NASA CONTRACTOR REPORT

NASA CR-876

A DESCRIPTIVE MODEL FOR DETERMINING OPTIMAL HUMAN PERFORMANCE IN SYSTEMS VOLUME I

PART A

A SIMPLE MODEL OF A MAN-MACHINE DEVELOPMENT CYCLE

by J. Jepson Wulff and Jon N. Leonard

PART B

A SIMPLE CALCULUS FOR DISCRETE SYSTEMS

by J. Jepson Wulff, Alden F. Pixley, and Jon N. Leonard

Prepared by
SERENDIPITY ASSOCIATES
Chatsworth, Calif.
for Ames Research Center

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION • WASHINGTON, D. C. • JANUARY 1968

NASA CR-876

A DESCRIPTIVE MODEL FOR DETERMINING OPTIMAL HUMAN PERFORMANCE IN SYSTEMS

VOLUME I

PART A
A SIMPLE MODEL OF A MAN-MACHINE DEVELOPMENT CYCLE

By J. Jepson Wulff and Jon N. Leonard

PART B
A SIMPLE CALCULUS FOR DISCRETE SYSTEMS

By J. Jepson Wulff, Alden F. Pixley[1], and Jon N. Leonard

[1] Assistant Professor of Mathematics, Harvey Mudd College, Claremont, California.

Prepared under Contract No. NAS 2-2955 by
SERENDIPITY ASSOCIATES
Chatsworth, Calif.

for Ames Research Center

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

For sale by the Clearinghouse for Federal Scientific and Technical Information
Springfield, Virginia 22151 – CFSTI price $3.00

A DESCRIPTIVE MODEL FOR DETERMINING OPTIMAL
HUMAN PERFORMANCE IN SYSTEMS

Volume I

PART A

A SIMPLE MODEL OF A MAN-MACHINE DEVELOPMENT CYCLE

PART B

A SIMPLE CALCULUS FOR DISCRETE SYSTEMS

Volume II

PART A

SYSTEM DEVELOPMENT ACTIVITIES CONCERNED WITH
PUTTING MAN IN AN AEROSPACE SYSTEM

PART B

DEVELOPMENT OF MAN-MACHINE SYSTEMS:
Some Concepts and Guidelines

Volume III

AN APPROACH FOR DETERMINING THE OPTIMAL ROLE OF MAN
AND ALLOCATION OF FUNCTIONS IN AN AEROSPACE SYSTEM

Volume IV

A DESCRIPTIVE MODEL FOR DETERMINING OPTIMAL
HUMAN PERFORMANCE IN SYSTEMS
Final Summary Report

FOREWORD

This report presents a model for predicting and controlling the course of a complex aerospace system development cycle to the end that the system which is produced will include man in an optimal manner.[1] The research was sponsored by the National Aeronautics and Space Administration, Ames Research Center, under Contract NAS 2-2955.

This effort was greatly enhanced through the interest and support of the technical monitor, Mr. Charles Kubokawa of the Biotechnology Division at Ames Research Center. Acknowledgment is also due to Professor Warren E. Wilson, Chairman of the Engineering Department, Harvey Mudd College, and to Dr. Elliot Axleband, both of whom reviewed early versions of the development cycle model and made important contributions to its improvement.

[1] Part B presents a simple calculus for discrete systems.

CONTENTS

PART A

PART B

PART A
A SIMPLE MODEL OF A MAN-MACHINE DEVELOPMENT CYCLE

CONTENTS

PART A

FIGURES

I. AN OVERVIEW OF THE REPORT

In the course of conceiving and developing an aerospace system, there is frequent need to be able to predict the necessary development events that have not yet taken place. For example, many times during development it is necessary to estimate the cost of that part of the development cycle which remains. In order to estimate cost, ordinarily one must first predict the things which remain to be done. For another example, early in the business of system development it is necessary to predict, and thus to prepare for, the manning and equipage of the design and production teams. For yet another example, it is often desirable to be able to predict the supporting research and development activities that must be carried out to provide the data inputs necessary to enable development to progress in a timely manner. In fact, it is possible to continue to list many examples of occasions which require the prediction of the course of events that a development cycle must, or should, follow between some present time and completion.

In the past, we have successfully developed many aerospace systems. If it is true that such predictions as those exemplified above are required in the course of any development cycle, then how has the job of prediction been handled in the past? The answer is that we have employed crude models based upon our understanding of some of the commonalities which exist among development cycles. From one development cycle to the next there is much that is common, both with respect to the things that are done and the order of doing them. Thus, a development cycle for an automobile and one for an aircraft both incorporate design phases, and in both design precedes fabrication. If one restricts his concern to development cycles for aerospace systems, the number of events in common is even greater than if one wishes to compare very different man-machine system development cycles. Many of the important features that are common to all aerospace system development cycles will be discussed later in this report. Because there are common features, someone who is familiar by experience with past aerospace system development cycles is able to set down (either implicitly or explicitly) a sort of model for a typical aerospace development cycle. Most predictions that are made today are based upon such models which are generated out-of-hand in order to

satisfy a present need for a model that will enable prediction. It is necessary today to employ models which are generated in this manner because there is no satisfactorily detailed, documented model which is useful for the purpose of prediction. To date, no satisfactory model has been set forth in the open literature where it may be subjected to examination and comment, and eventually to gradual improvement toward one that will truly meet the needs of managers of aerospace system development cycles. That is not to say that the literature is completely devoid of consideration of development cycles. The Air Force, for example, has prepared incredibly detailed documentation of many of the features of the development cycle which it desires to be employed in the development of its own military systems. The Air Force documentation (ref. 1, 2, 3, 4, 5, and 6) with all its detail is, however, tailored for the solution of specific Air Force management problems, for ensuring that the proper hierarchical structures are preserved during system development, and so on. One would have to perform a great deal of labor to abstract from the Air Force documentation a simple basic picture of aerospace system development that would serve the many needs for a device to enable predictions. Several authors in the field of system engineering, (including ref. 8, 21, 22, and 23) discuss the general problem of the system development cycle and present brief characterizations of the process. However, most characterizations fall far short of the detail needed, although at least one (ref. 8) approaches it and is supported by an appropriate rationale.

What is done today in the way of modeling for the purpose of enabling prediction is perforce makeshift. What is needed is a model in sufficient detail that it can be used to improve predictive processes. Clearly, there can eventually be developed alternative models which will rate "good, better, and best," but it seems prudent to expect that the first version will not be better than "good." Before a better one can be developed, a good one must be set forth in a public manner so that it can provide a stimulus for the development of a better one. If the first one has a sound basic structure, it will be possible for future efforts to build upon it. It is thus desirable that a model be produced for the use of those system managers who must make predictions about system development, and that every effort be made to provide this first model with a sound framework. This report presents a model which has been

developed in an attempt to satisfy this need. After consideration, the model offered may be useful for other purposes as well. It may, for example, be found to provide a useful basis for indexing the data needed in an aerospace system development cycle.

The focus in this report is a biased one. First, the report is focused upon aerospace system development cycles. Second, in developing the model we were more interested in serving the needs of system managers concerned with personnel products than the needs of hardware engineers. There is no reason to believe that the basic approach employed could not be used to extend the model to include matching detail with respect to the development of the hardware side of an aerospace system. However, for the present the model contains only as much detail about hardware development activities as needed to provide the context for detailed identification of development activities which deal with getting man into an aerospace system in a defensibly good manner. The model is thus designed primarily to satisfy a requirement for a model to enable personnel products-oriented system managers to make predictions about the personnel products facets of an aerospace system development cycle. It is specifically tailored to be useful in predicting (and controlling) the course of an aerospace system development cycle to the end that the system which is produced by it will include man in an optimal way. It is expected that this report will also be useful to anyone interested in the general problem of complex system development and to those responsible for the overall planning of specific aerospace system development cycles. The model has been developed by employing a simple system calculus the basic concepts of which are presented in Part A. The calculus is fully presented in Part B of this report, A Simple Calculus for Discrete Systems.

There are two other related reports which may be of interest to the reader of this report. Report IIA, System Development Activities Concerned with Putting Man in an Aerospace System, is a detailed consideration of the activities in the development cycle model which have to do with putting man in the system under development. The information presented in this report is sufficient to provide support for planning and controlling such activities in an aerospace system development cycle. Report III is entitled, An Approach for

Developing the Optimal Role of Man and Allocation of Functions in an Aerospace System. This report contains information to support the processes of determining man's role in an aerospace system and of allocating to man just those duties and tasks that he should be required to perform to develop an integrated man-machine system which is as a whole "optimum." Report IIB, Development of Man-Machine Systems: Some Concepts and Guidelines, is intended for use by anyone who desires to understand the usage and implications of concepts commonly employed in talking about man-machine system development. This report not only discusses common concepts and terms in the jargon, but also relates them to the concepts and terms chosen for use in the other reports in this series.

The model that is given here in Report IA is presented and explained by means of conventions which are not broadly employed, and its development has been based upon assumptions and concepts which should, in all fairness, be made known to the reader. Therefore the following section presents to the reader these conventions and assumptions. It is believed that this section will provide the reader with a good basis for undertaking to use and improve the model. Following the establishment of this groundwork the model is presented in symbolic form along with its rationale.

The report is terminated with a section which discusses how the model may be used. There is an appendix which reports the method by which the model was developed. By making this method a part of the public documentation we hope to serve students of the system development processes who might be inclined to retraverse old ground.

II. CONVENTIONS AND ASSUMPTIONS

There is a need for a technology of development cycles. However, it is clear that such a technology cannot be developed in one big step. As in the case of other branches of science it may be expected that the technology of development cycles will evolve over time as interest in the topic is sparked by need and progress. To enable evolution and the benefits which evolution brings, there must be a way to communicate about development cycles publicly and without ambiguity. Thus, to evolve, a technology requires its own public terms, symbols, conventions, and concepts. Present ways of talking about development cycles are, for the most part, not suited to the promotion of this evolutionary process. The words which are currently used are simply too ambiguous, too rich with alternative implications, to be used in successful communication. That this is so may be seen by examining a companion report, Report IIB, Development of Man-Machine Systems: Some Concepts and Guidelines. Suffice it to say here that profitable, public, and precise discussion about the topic of development cycles requires that the terms to be employed first be fully exposed for the consideration of the participants. To this end, this section describes the manner in which some forty special-purpose terms are employed in presenting the development cycle model which is the principal topic of this report. Some of the words discussed are old words familiar to the reader. These are presented here primarily to ensure that the implications of the terms are indeed public. Many of the terms, however, are new ones, not in common use. These terms have been introduced because there is no satisfactory term in the vernacular for the concepts to which they refer, or because the words in the vernacular which might be used appear to have so many alternative meanings that they might give rise to difficulty of interpretation on the part of the reader. What we say here about these terms is not necessarily "right"; it is merely what has proven to be useful thus far in the pursuit of our objective.

The first half of the list of terms discussed below provides a formal way of talking about any discrete system.[1] Such a formal method is needed here because in this report we will treat a development cycle as a discrete system. Several of the terms needed in order to talk about development cycles as discrete systems are precisely defined in Part B of this report, A Simple Calculus for Discrete Systems. These terms are given working definitions in this chapter. They are: state, function, partitioning/adding, compound inputs, system, monitoring functions, additive loop.

The calculus for discrete systems was devised to enable the modeling of aerospace system development programs in accordance with a public method; it was not devised for the specific purpose of describing the operational aerospace systems produced by such development programs. To be used with consistency for modeling all kinds of operational aerospace systems, the calculus would have to be modified. Nevertheless, several of the concepts in the calculus and several concepts related to its application are employed in this report for the purpose of talking about operational aerospace systems. It is believed that these "extensions" in use are fruitful and will prove to be fair. Unfortunately, these "extensions" cannot now be defended rigorously.

The second half of the list of terms beginning with Primitive Need Statement is made up of terms which are specific to the business of development cycles. These words are needed in this report in order to give meaning to a formal model of a development cycle in the real world of aerospace system development.

The order of presentation of the terms in this section has been chosen to permit the use of early terms in the discussion of those which are presented

[1] A discrete system is one whose operation can satisfactorily be described as a sequence of events moving forward in time and whose terminal output state is fully described at a point in time after which no further events occur. Such a system must be one whose condition at any point of time can be satisfactorily described by stopping the clock and by identifying the complete condition of the system at that point in time. For a more complete definition, see Part B of this report. See also the definition of system which follows in this section.

later. Therefore, the reader may find it difficult to read the following discussions out of order.

State

The special meaning adopted here for the word state underlies the meaning of virtually all of the special terms that will be used. In these reports, state is defined to include three things: (1) a symbolic statement, (2) a specified act of measurement, and (3) a point in time. Before attempting to give a complete intuitive definition of the word, it will be useful to explore what is meant by symbolic statement and act of measurement.

By act of measurement we mean any public prescription for reliably[1] making a class of measurements of the real world. To give an example, the act of measurement involving the use of a meter stick to measure length is a public act of measurement, the meter stick is defined by a standard which is publicly available, and the manner of its use is also publicly prescribed and generally employed. An act of measurement is thus a formula which can be used by anyone.

The result of an act of measurement is a symbolic statement, for example, "2 meters." Thus a symbolic statement may be made up of a number such as two and a unit of measure such as meter, but it need not always involve number. For example, by performing acts of measurement it is determined whether or not to apply the symbolic statement "sophomore" to an erstwhile freshman.

In this report, we will assume that an act of measurement occurs at a point in time. The effect of performing one specific act of measurement is therefore to place a symbolic statement in correspondence with that part of the real world which has been measured at a point in time.[2]

---

[1] We need not be concerned here with the question of validity. An act of measurement is public when it can be carried out with reliability by any trained person. The concept under discussion is, of course, related to the concept of operational definition (ref. 11).

[2] In order to permit use of the concept of state in the design of systems, we will permit the act of measurement to take place at a point in time in the

Although the concept of state includes a symbolic statement, an act of measurement, and a point in time, in practice a specific point in time need be identified only when there is need for that information. [1] Thus, in general, it is not necessary to denote a state as in the following example: (12.3 gallons, British volume measure, Noon UT 23 January 1942). Usually, it will be entirely satisfactory to denote a state without identification of a specific point in time but with the understanding that there is a point in time at which measurement was or is to be taken. For example, it may be entirely satisfactory in a given case to denote a state as, for example, (tubercular, patch test). In this case it is understood that the symbolic statement, tubercular, is in correspondence with a person at a point in time. It is implied that it is not important in the use situation specifically to identify the point in time.

In common practice, we often denote a state by making a symbolic statement such as "10 centimeters" dropping not only notation of time but also specific notation of the act of measurement. This practice is permissible whenever one can infer the act of measurement without ambiguity; however, in some cases one cannot. For example, the symbolic statement "100 feet of altitude" may be quite unsatisfactory unless the act of measurement is identified. Thus, if the act of measurement employs sea level as zero altitude, the implication is different from the case in which ground level is taken as zero altitude. It is because of the possibility of this kind of ambiguity that a state is defined to include both a symbolic statement and identification of the act of measurement.

In sum, it may be satisfactory to denote a state as, for example, (100 milliliters) when there is no ambiguity with respect to the act of measurement nor need to identify the time of measurement. It is understood that this symbolic statement is in correspondence with a part of the real world at a

---

future. This will enable us to talk about system states before the system is fabricated.

[1] We choose to include the notion of measurement at a point in time in defining state because the concept of state which results is entirely adequate for our purpose and avoids many difficulties that would be encountered if we chose to include measurement over an interval of time.

point in time, but that the specific point in time is not a useful element of information and that the act of measurement is the commonly used act of measurement of length which results in milliliter units.

To complete our intuitive definition of the term state, we need make only one more extension. We may conceive of two or more acts of measurement being carried out at the same point in time such that we generate a set of symbolic statements each of which has its own act of measurement but all of which are made at the same time. We shall also call such a set of states a state. Thus, we will employ the word state to refer to a set, and it will be understood that sometimes the set may have only one member, but that at other times it will have several members which qualify for inclusion in the set because they have in common the same point in time.

Function

Our definition of state as a symbolic construct anchored to the real world by means of an act of measurement provides us with a basis for developing other symbolic constructs by employing state as a building block. Function is such a construct.

A function is defined as a pair of ordered states (a, b) in which the time associated with the first state, t_a, is earlier than the time associated with the second state, t_b. The definition of function also includes a table which details information about probabilities of states in the function. Thus, to be complete a function would be defined as (for example):

Function A = (a, b) such that

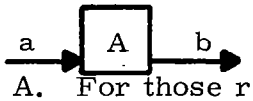
Given prob. of input = 1 for:	Prob. Output
a	P_A
ȧ (not a)	0

The first coordinate in a function will be referred to as an input state or simply as an input, and the second coordinate will be referred to as an output

state or output. The probability of the second coordinate as given in the table will be referred to as the probability of output.

It should be remembered that state has been defined as a set and that both the input and the output of a function are therefore sets.

Properly used, in the context of these reports, the word function refers to a construct; it never has a referent in the real world (see the discussion of means which follows).

We will frequently use the symbol 

to denote the function (a, b), or more simply, the function A. For those readers who may be accustomed to using "boxes and arrows" in other contexts, a word of clarification may be in order. In using boxes and arrows to denote functions, arrows will denote states, and therefore there is a single point in time associated with an arrow. Intuitively speaking, all points on an arrow will be at the same point in time and all "changes" will be associated with boxes.

Unless otherwise stated, it may be assumed for any function (a, b) that the probability of the output state will be zero, given a. Thus, the second line in the probability table above may be assumed. Under these circumstances, all of the probability information needed to define the function (a, b), for example, may be given by a value for P_A, the probability of the output state, b, given a. Therefore if P_A is given as 0.8, the following table is implied:

Given prob. of input = 1 for:	Prob. of Output
a	0.8

Means

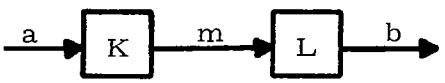
Any real world process which may be put agreeably in correspondence with a symbolically defined function is a means. Thus, to place a process in correspondence with a function, the sets of measurement in the input and output states of the function must be performed.

Real-world processes typically involve objects and the word means is therefore used also to refer to the objects which have latent capability to carry out a process described by a function. Thus, in a sense, means are to functions as real-world attributes at a given point in time are to symbolic statements.

In the design business, one is characteristically concerned with the task of identifying functions for which implementing means can then be found. Thus, in design, one works from a symbolic statement of a function to the identification of a means. In general, a designer working at the functional level of design is interested only in functions which can be implemented. Frequently, a function can be implemented by several alternative means. The function-to-means relation is therefore a one-to-many relation in many cases. It is interesting to note that the means-to-function relation is also frequently a one-to-many relation. Thus, given an object (means) one can frequently find many functions which it can implement. This inverse relation is not often of interest to the designer (see allocation of functions later in this section).

Partitioning/Adding

For any function there is a large number of arrays of two or more functions that are proper partitionings of the function. Thus, for example, a typical partitioning of the function (a, b) is:

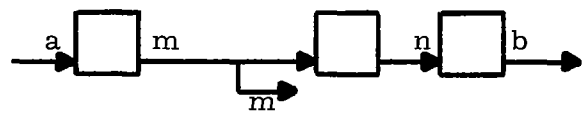


When an array of functions that is a proper partitioning is substituted for a given function, the function is said to be partitioned. The functions in the partitioning are said to be component functions of the parent function.

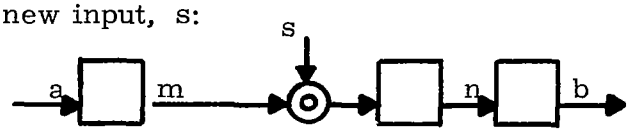
When the function (a, b) is partitioned into functions (a, m) and (m, b), the probability of output b must remain the same as in the table for the parent function, (a, b). Thus, the tables for functions (a, m) and (m, b) must be such that the product of P_K and P_L will be P_b.

The variety of ways in which a function may be partitioned is discussed in the companion report, A Simple Calculus for Discrete Systems. The reader who is interested in rules for discriminating improper from proper partitionings should refer to that report.

Two important rules are: (1) partitioning may not expose any new system outputs (outputs earlier in time than the terminal output of the system), and (2) partitioning may expose new system inputs so long as the probability table for the parent function is not violated. Thus, the following is an improper partitioning of function (a, b) because it exposes a new system output, m, earlier in time than b.

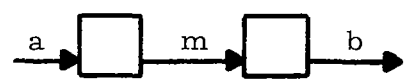


On the other hand, the following partitioning may be a proper one even though it exposes a new input, s:



(See the following discussion for the meaning of the AND symbol, ◎.)

The inverse of partitioning is adding. All proper partitionings are reversible. Thus one may add:

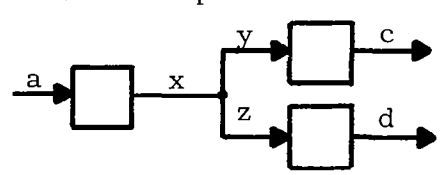


to obtain the function, 

.

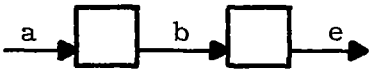
Compound Inputs (ANDed Inputs)

Occasionally it will be useful to partition a function (a, b) in the following manner:

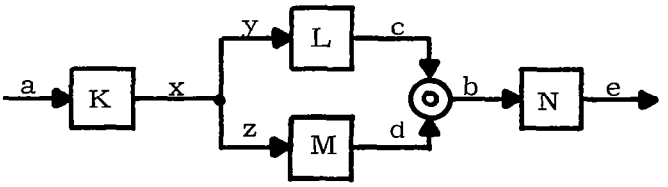


In order that the implication of the notation in the diagram above will not be ambiguous, it is important to note that it does not imply a "splitting" of state x at the dot into states y and z. Rather, it may be inferred from the diagram that the set x contains the subsets y and z. Of the elements in the set x only the elements in the subset y are in the input state to the function (y, c) and only elements in the subset z are in the input state to the function (z, d). Further, if the above is asserted to be a proper partitioning of the function (a, b), then the output state b is seen to contain the subsets c and d.

Now let us consider the partitioning of the following array of functions.



If we partition the function (a, b) as before, but do not partition the function (b, e) then we write the partitioning:



As it is used in the above figure, the symbol ◎ implies that the input state for the function (b, e) on the right is the set of states b which is made up of subsets c and d such that neither c alone nor d alone is an input to this function. The and symbol, ◎ , calls attention to the fact that the input state to the function (b, e) is compounded of the subsets c and d.[1]

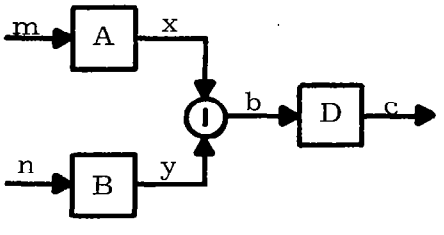
[1] In the above example, if $x = y = z$, then $P_x = P_y = P_z$. If $x \neq y \neq z$, then P_x, P_y and P_z must all be given, P_y and P_z must be reconcilable with P_x, and the values given must combine with values for P_L and P_M to yield P_b as given in the original table for function (a, b). Also, values for P_L and P_M must be such that $P_c(P_d) = P_b$.

The AND symbol refers to the input state of the function to its right: it has no implications with respect to the functions that precede it.

OR Inputs

It will sometimes be useful to employ the concept of OR inputs. In the diagram below, the symbol ⓘ is the OR symbol. It denotes that the table for Function D may be written in terms of b or in terms of x and y. Further it denotes that the table may be according to the following model:[1]

Given:	Prob. Output
x and $\dot{y}$	P_D
$\dot{x}$ and y	P_D
x and y	O
$\dot{x}$ and $\dot{y}$	O



The OR symbol does not imply a function. Thus, $t_x = t_y = t_b$.

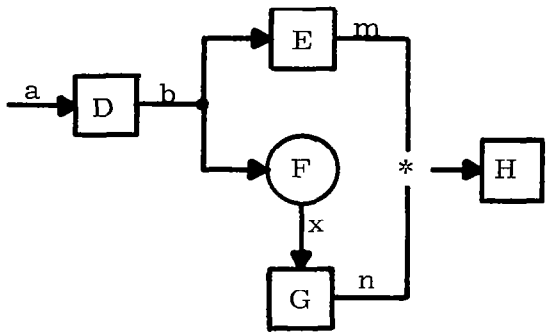
[1] The OR concept presented here may be recognized as the exclusive OR. The nonexclusive OR will not be used in this report. It may be defined and employed in the calculus, however. It is needed when one wishes to model the simple parallel redundant situation.

The probability of output b, is given by:

$$P_b = P_x (P_{\bar{y}}) + P_y (P_{\bar{x}}) \text{ or}$$

$$P_b = P_x (1 - P_y) + P_y (1 - P_x)$$

The OR symbol ⓘ always refers to the function to the right; the symbol does not refer to a characteristic of the array of symbols to the left. Thus, in the array given below, (m and n) will never be seen because the Functions D, E, F, G are configured in a manner to preclude (m and n). In this case, the Function H may not have OR characteristics. When it does not, an asterisk and a note may be employed to remind the reader that m and n are alternative output states. (Function F in the diagram is a monitoring function; this type of function is discussed later in this chapter.)



* D, E, F, G is configured such that m and n are alternative states.

System

The term system is a term of convenience. Thus, in a given discourse involving the partitioning of a function it is frequently useful to have a special term to identify the "parent" function. The name which is used for this purpose is system. Thus, by "the system" we mean the function which encompasses all of the other functions in the discourse that are not specifically identified as lying outside of the function called the system. System is also used to refer to any complete array of component functions of the function named "the system" in the discourse.

It is important here to recall that it is not our purpose in this section to attempt to define terms such as system in order to account for common usage of them. Rather, our purpose is to give precise meaning to selected terms so that they may be employed without ambiguity in this report. It is recognized, therefore, that system is employed to mean many different things in everyday discourse.

In addition to its use in formal discussions of functions, the term system is also used to refer to a collection of means which may be set in correspondence with the function which has been earmarked as the system. Typically it will be possible to infer from the context of a discussion when the word system is being employed to refer to a special function, and when the term system is being employed to refer to a physical system.

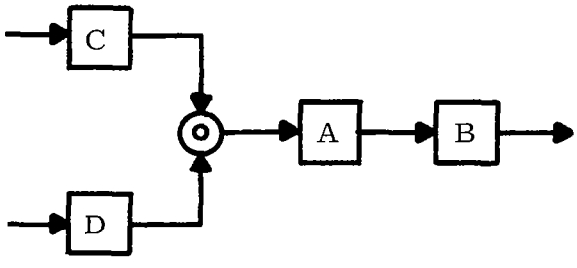
Follow-On System

A system may be identified as a follow-on system (or follow-on function) only in a relative sense. Thus, in the array of systems below, system B is the follow-on system to system A.

Given any system (A), the systems (B) which receive as inputs the output of A are called follow-on systems. A system may have more then one follow-on system.

Adjacent System

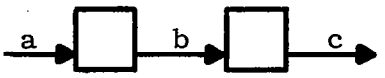
This is yet another term of convenience. In any array of related functions, the systems which provide inputs to a given system are said to be adjacent to it. Thus, in the array below, the adjacent systems to system A are systems C and D. When the term adjacent system is employed to refer to real-world systems, it has additional implications. These are discussed below under the heading, "A" Score Formula.



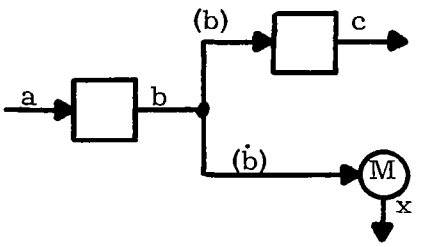
Monitoring Function

As the function (a, b) is defined, "not b" (symbolized $\dot{b}$) is the complement of the output state b, and $\dot{a}$ is the complement of a.

Let us consider a function (a, b) and its follow-on function (b, c).



In the above array, the output of the first function is the input to the second. New let us insert another kind of function, one which responds to $\dot{b}$:



The table for Function M is:

Given Input	Prob. of Output
$\dot{b}$	P_x
b	0

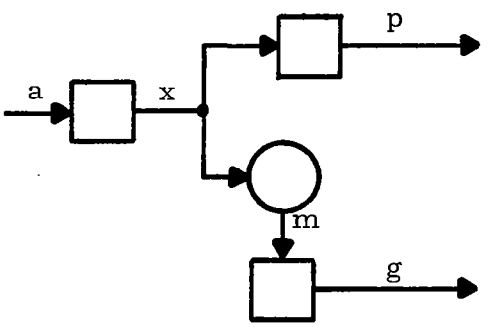
The function ($\dot{b}$, x) in the diagram above is called a monitoring function. A monitoring function is one whose input is the complement of an output state of a function. The insertion of a monitoring function into an array of functions creates a second system output. An array which contains a monitoring function with a disconnected output is not a permissible partitioning. [1] However, when the output state of a monitoring function is articulated with other functions

[1] A monitoring function on the output state of a system is the single exception.

such that no secondary output appears, then the inclusion of a monitoring function may be permissible. For example, the function (a, f) whose table is:

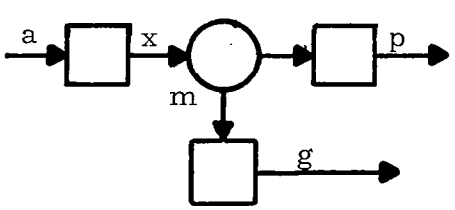
Given Input	Prob. of Output
a	.8
ȧ	0

May be partitioned:



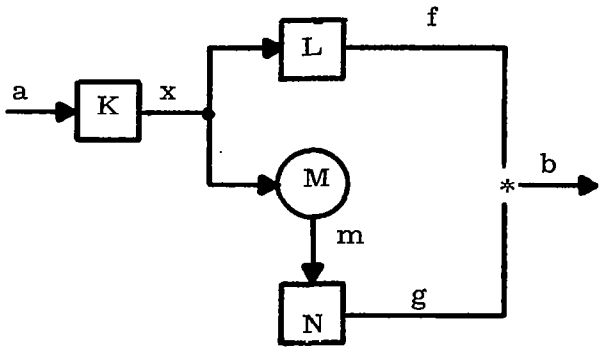
(p and g are subsets of f)

A monitoring function is symbolized in a function diagram by a large circle as demonstrated in the figure above. Whenever a function is symbolized by a circle, it is a signal that the input state is the complement of the state given in the diagram as the output state of the adjacent function. Sometimes a monitoring function will be shown in a system diagram in the manner indicated below simply because this shorthand is more convenient.



Additive Loop

The function group below is a frequently used partitioning of the function (a, b).



* The functions K, L, M, N are configured such that when f is seen, g is not, and when f is not seen, g may be seen; b will therefore be f or g, but not both.

The probability expression describing this array is:

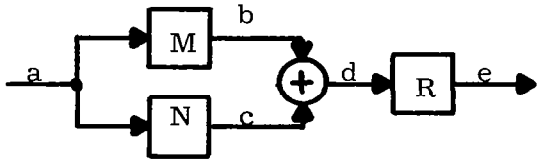
Given $P_a = 1$

$$P_b = P_K\ (P_L) + \left[(1 - P_K)\ (P_M)\ (P_N)\right] = P_f + P_g$$

It can be seen that only one of the additive terms above includes the monitoring function, M. Any array of functions which is represented as an additive term in the probability expression describing a system and which includes a monitoring function is called an additive loop. [1]

---

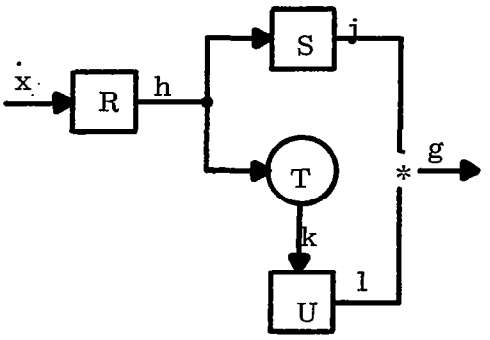
[1] All nonprime additive terms except those which express parallel redundancy include a monitoring function. True parallel redundancy is expressed symbolically as



the ⊕ implies that the table for function (d, e) may be written in terms of b or c or (b and c) as inputs. Thus $P_d = P_c\ (P_b) + P_{\dot{c}}\ (P_b) + P_c\ (P_{\dot{b}})$. The term prime is defined in a following discussion.

The elements in any additive loop may be added and expressed as a function. Thus, M and N may be added and expressed as the function $(\dot{x}, g)$.

The function $(\dot{x}, g)$ may in turn be partitioned to include its own additive loop. Thus:



* R, S, T, U is configured such that j and 1 are not seen simultaneously

In the above example T and U are an additive loop. They will be referred to as a second-order additive loop.

It can be seen that by using additive loops one can functionally describe such real-world means as standby redundant hardware and repair actions, both of which are carried out to add to total system probability of output.

Additive Set

The set of all additive loops in a system is referred to as the additive set. It will only be under very rare circumstances that the additive set may be called a system, for the additive loops in the additive set normally account for many different output states occurring at many different points in time.

Recognizing that there may be orders of additive loops, we may also refer to the first-order additive set, to the second-order additive set, and so on.

The first-order additive set, for example, will contain all of the first-order additive loops in a system. When the term additive set alone is employed, it is intended that all orders of additive sets be included.

Prime System

When the functions which result from the partitioning of a system are related in such a manner that the associated probability equation contains only factors, then the array of functions may be called a prime system. Any system may be partitioned into an infinite number of different prime systems. Sometimes it is desirable to pick one of these prime systems as a key prime system and to call it the prime system. Any prime system may be picked as the prime system.

Model

We will use the term model to refer to any set of symbols or objects which can be placed usefully in correspondence with a real-world system of interest according to some public rule. We exclude only a collection of objects that can be a true replacement for the system of interest. Such a collection of objects is more properly referred to as a copy of the system.

Frequently used system models include engineering drawings, parts lists, wiring diagrams, assembly diagrams, function diagrams, computerized simulation models, mock-ups, and scale versions of the system.

Primitive Need Statement

A Primitive Need Statement is a statement in any language, technical or nontechnical, which calls attention to a problem in a system of concern to the person who utters it. There is no prescribed "form" for a Primitive Need Statement because it is the first statement which calls attention to a problem and is therefore frequently uttered by a layman who cannot be bound by any prescribed form. Such a statement serves its purpose when it sets a chain of activities in motion directed toward doing something about the problem;

a statement which results in no such activity will be lost and will never be identified as a Primitive Need Statement. A Primitive Need Statement may fail specifically to identify the problem; it will be sufficient if it calls attention to the fact that there is one, and if it then results in activity directed toward solving it. The problem to which attention is called may be a problem in a man-made system, or in a natural system. If the system of concern is a man-made system, it may even be one which is still in the design stage.

The term Primitive Need Statement, is borrowed from a useful book by Morris Asimow. Dr. Asimow states:[1]

> The starting point of a design project is a hypothetical need which may have been observed currently on the socio-economic scene. It may be phrased in the form of a primitive statement resting on untested observations; or it may have been elaborated into a sophisticated and authenticated statement based on market and consumer studies. The need may not yet exist, but there may be evidence that it is latent, and that it can be evoked when economic means for its satisfaction become available. The need may be suggested by a technical accomplishment that makes the means of its satisfaction possible.

Customer

We will use the word customer to refer to a person or agency who is responsible for a system-with-a-problem that gives rise to a Primitive Need Statement. If the chain of activity initiated by a Primitive Need Statement gives rise in turn to the design, fabrication, and installation of an operational system, then the customer is the person or agency responsible for the follow-on system to the operational system that is installed.

The concept of a customer is important in system development because it identifies the individual with the right and responsibility to decide whether or not his system has a problem, and to decide when or to what extent that problem is solved when an effort is made to solve it. Recognition of a customer

---

[1] Introduction to Design, page 18. (ref. 8).

precludes a problem-solving effort which sets its own criteria for success and which is the ultimate judge of itself. Recognition of a customer also precludes expanding a development effort ever forward so that it encompasses more and more of the problems in the chain of systems ahead of it.

It should be noted that the customer is not always the funding agency for a development project. In many cases, the funds for developing, installing, and operating an operational system which solves a problem in a follow-on system will be a third party—a government agency interested in both the operational system and the follow-on system, for example. Care should be taken not to ascribe to a pure funding agency the rights and responsibilities of the customer. On the other hand, a funding agency which supports both the operational system and the follow-on system should have interest in securing a relationship between the two that would have overall optimum cost-effectiveness.

Need Satisfaction Score

Let us identify the system of concern to a customer as system B. When we talk about a problem in system B (as in a Primitive Need Statement), we ordinarily mean that the output of system B is inadequate according to some criterion. Thus, a problem in system B typically can be identified only by its undesirable symptoms. The undesirable symptoms (outputs) may be system outputs that are not satisfactory to the follow-on system of system B, or they may be spurious outputs of system B which have undesirable effects on other systems in its environment.

When a Primitive Need Statement is uttered about a problem in system B, the person making the Primitive Need Statement may have responded to any one of a large variety of types of cues. For example, he may have observed the system output of system B to be out-of-tolerance, or he may have observed that a spurious output of system B was having a bad effect on some other system in its environment, or he may have observed that some physical part of system B was broken, or he may even have observed that the cost of operating or maintaining system B has been too high. Whatever the observation, if action is justified to follow it up, analysis of system B must lead to the identification

of an output which has an undesirable effect upon its follow-on system or upon other systems in its environment. If no such bad or undesirable effects can be identified, then the expenditure of resources to correct a hypothetical "problem" could not be justified.

We may say then that a valid Primitive Need Statement will lead to analysis of system B, and that analysis will identify a measurable output of system B that is for one reason or another undesirable. Sometimes, a bad output of system B will not be measured directly, but will be inferred from the observation of an out-of-tolerance output of a component function of system B which can logically be shown to cause a bad system output. When analysis reveals a bad measurable output of system B or a bad measurable output of a function of system B, then it may be said that the problem underlying the Primitive Need Statement has been located. (It may be, of course, that several problems will be located as the result of analysis. None of what follows in this discussion will be greatly changed in this case, but to consider the possibility of several simultaneous problems throughout the discussion would unnecessarily complicate the following discourse.)

When the customer agrees that a problem of concern to him has been located as the result of an analysis following a Primitive Need Statement, it will be desirable next to develop a measure by which the magnitude of the problem may be assessed. Thus, it will be desirable to find a way of measuring the undesirable output (either directly or indirectly) that will yield scores which identify the relative "severity" of the problem as the severity changes. For example, a desirable range of scores might be from zero to one such that <u>one</u> would indicate complete solution of the problem and such that <u>zero</u> would indicate no solution. The formula (or act of measurement) for obtaining such scores would assign numbers between zero and one to enable the ranking of less than perfect solutions to the problem. Such a formula for measuring the "size of the problem" will be called a Need Satisfaction score formula. Thus, a <u>Need</u> Satisfaction score formula is one for measuring the extent to which the <u>need</u> of the customer is satisfied when an attempt is made to solve his problem. The construction of a Need Satisfaction score formula does not of itself have any corrective effect; it merely provides for an objective way of

measuring and comparing alternative methods of solving the problem. It provides a method of measurement fully supported by the customer which can be the basis for identifying a target Need Satisfaction Score. The implication of a target score is that any candidate solution for the problem which yields a lesser score will be completely unacceptable.

It should be especially noted that a Need Satisfaction Score formula is never tailored for the measurement of a particular method of solving the problem of the customer; it is always unbiased so that it may be used to evaluate any proposed solution. The basic technique for obtaining an unbiased method of measurement is to base the formula solely upon consideration of system B, the system of the customer, and assiduously to avoid consideration of any means by which the problem might be solved while generating the Need Satisfaction Score formula. When composed in this manner, a Need Satisfaction Score formula may be employed as an agreement between the customer and someone hired to solve his problem as to how they will determine when the problem has been solved.

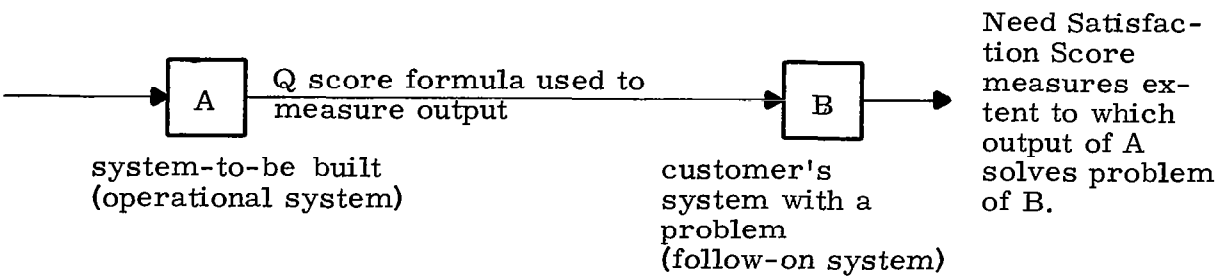
Quality Score

In this report, we are concerned with needs or problems which require complex systems for their solution. This is not to say that complex systems are favored; rather, we merely wish to note that this report is not specifically designed to be useful in those cases in which there is a simple solution to the problem identified by a Need Satisfaction Score formula.

We are concerned here with the case in which we must build a complex system, A, in order to achieve a target Need Satisfaction Score in its follow-on system, B. In general, if system A must be complex and costly, and can be justified, then the problem in its follow-on system must be one of significant importance to society.

Given a Need Satisfaction Score formula and a target score for system B, we have a criterion for the success of system A, but we do not have a specific identification of its output boundary. A way to make such a demarcation must be provided.

One way to obtain an identification of the output required of system A is to model system B [1] in such ways that one may explore the effects of various hypothesized inputs upon its Need Satisfaction Score. The results of such an exploration can be the development of a formula for directly measuring a hypothetical output of system A in such a way that the resulting Need Satisfaction Score can be predicted purely on the basis of output measurement of A.



We will call a formula for measuring the output of system A in such a manner that the resulting Need Satisfaction Score can be predicted, a Quality score formula. It will be useful to think of Quality scores which result from the application of this act of measurement as falling in the interval zero to one, in the same manner as Need Satisfaction Scores. Thus, a Quality score of zero will correspond to a Need Satisfaction Score of zero and a Quality score of one will correspond to a Need Satisfaction Score of one. By means of this device, we provide a way of identifying objectively the output needed from system A without entangling ourselves in the inner workings of system B and without necessitating the use of the Need Satisfaction Score. Thus, a target Quality score which corresponds to a target Need Satisfaction Score will establish the lower boundary of acceptable output of system A, and we may then say that system A may be implemented in any manner that will yield a Quality score greater than or equal to the given target Quality score.

[1] If system B is an electronic system, for example, there will probably exist a model of the system which will readily permit the development of a Quality score formula. On the other hand, if system B is a natural system, such as the human physiological system, then development of a model for the purpose of deriving a Quality score formula may be difficult indeed. In some cases, in fact, it may not be possible to test the goodness of a hypothesized relation between the Quality score formula and the Need Satisfaction Score formula until after system A has been built and tested in conjunction with

For complex systems, the makeup of a Quality score formula can itself be very complex. Whatever its makeup, it is clear that it must be determined by consideration of system B and not by consideration of system A means, for system A will not exist at the time of Quality score formulation and, in fact, wants defining until the Quality score formula is prepared. Inasmuch as a Quality score formula is constructed on the basis of an analysis of system B, there is no single prescription for what it must include; it must be tailored to the system B at hand. The best that can be said about the content of a Quality score formula is that it will most likely include provision for the measurement of a number of factors and provision for combining the obtained factor scores into a single overall Quality score. Some of the factors which must be taken into account will include: the output state, the time when the output must first be made available, the life required of system A, the dependence of the output on external signals, the probability of output required, and the conditions of use under which the output must be provided.

In practice, it may be very difficult or even impossible to prepare a Quality score formula of the type described here and to obtain customer agreement upon it. Nevertheless, the eventual test of any implementation of system A will require measurement according to a formula. It can be seen that whatever formula is used for measuring system A, it will be used in exactly the same sense as the Quality score formula. It is asserted here that the method or formula by which Quality is measured should be public. Then whoever uses a Quality score will at least know what it means, even though he may prefer an alternative formula.

Operational System

In the discussion above, the system with the problem was identified as system B, and the system called out by the Quality score to solve the problem was identified as system A. Systems in the role of system A will be called

system B. But whether a specific Quality score formula can be well justified or not, it is clearly required as a device for describing objectively the output boundary of system A. Without a Quality score formula, there is no objective way of providing for a test of any proposed implementation of system A independent of system B.

operational systems. Systems in the role of system B will be called follow-on systems in accordance with the earlier definition of follow-on system. An operational system may have several follow-on systems at once. Thus, some operational systems provide an output which is employed simultaneously by two or more follow-on systems. For example, data gained on the surface of the moon may be needed simultaneously in the scientific community for strictly scientific purposes and in the engineering community for the design of vehicles to undertake further exploration of the moon. Whether there is one follow-on system or several, the output of an operational system is always identified by a single target Quality score with an associated Quality score formula.

Design Solution

Taken together, a Quality score formula and a target Quality score provide a single function definition of a needed operational system.[1] Thus, in effect, a problem is presented, "Find a collection of physical objects which can be placed in correspondence with this single function description of an operational system." A detailed description of an operational system that will solve the problem may be called a design solution. A design solution may be in terms of component functions or it may be in terms of means descriptions.

System Solution

The term design solution refers to a set of symbolic statements. It is useful also to have the term system solution to refer to a configuration of means that is a system which has been selected to solve a problem. For any given operational system defined by a target Quality score and a Quality score formula, there will be a very large number of system solutions. Some of the system solutions will have Quality less than the target Quality score, but none will have Quality equal to zero. Thus, a system solution for a given operational system may be defined as a collection of means with Quality greater than zero.

---

[1] Typically a Quality score formula will identify the input boundary of the system. When it does not, this statement is not strictly true.

One system solution is said to be different from another if they cannot be equated means-for-means at all levels of detail down to and including the most detailed level of manufactured components. Thus, if two implementations differ by a single rivet they will be different system solutions.

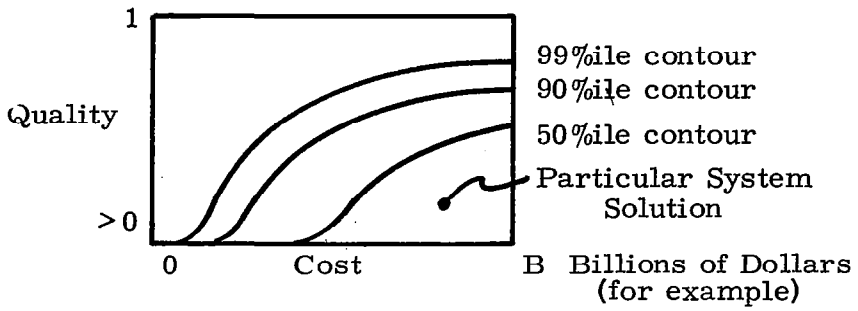
A design solution is a set of models that will always include many symbolic statements such that the statements taken together encompass the total system. Any set of states which includes one or more of the symbolic statements in a design solution will be called a design state. A design state need not encompass the total system; it may include only a small part of the system.

Cost

Every system solution for an operational system has cost associated with it. That is, in order to design, develop, install, operate, and maintain an operational system, resources must be expended. The totality of all of the resources required (in terms of dollars, materials, personnel, etc.) is referred to as the Cost of the operational system. In practice, for any given system problem there must be a specified formula for computing Cost. Ordinarily, such a formula will require that all needed resources be taken into account from the time of the Primitive Need Statement to retirement of the operational system. Needed resources may be described in monetary units, or they may be identified by category in units appropriate to each category. However they are identified, there should be a formula for computing and expressing Cost so that alternative system solutions may be compared in terms of Cost.

Cost, Quality Space

Any specific system solution for a given operational system can be characterized in terms of its Cost and its Quality. Thus, by virtue of its Cost and Quality attributes, a particular system solution takes a specific position in a Cost, Quality chart:



The chart above depicts what will be called the Cost, Quality space. The lower limit of the first coordinate is zero Cost; the upper limit is a very large resource cost beyond which consideration of Cost is not useful. The range of Quality scores excludes zero Quality; thus, the chart includes only system solutions. There is a unique Cost, Quality chart for each Quality score formula.

When a target Quality score is given as a lower bound of acceptable system solutions, and when an upper Cost boundary is given, the effect is to restrict consideration to solutions in the upper left-hand area that is bounded off. Usually, when a new system is called for, interest is restricted by some device to a selected portion of the space. Therefore it is of interest to know something about the distribution of solutions within the space.

We have already observed that the number of different system solutions for a given operational system will almost always be very large indeed. Therefore, there would be a very large number of points in the space, each representing a different system solution. We may further observe that the points will probably not be equally dense throughout the space. Our capability to design systems being what it is, it is always possible to find more systems on the relatively high-cost side of the space than on the low-cost side. And, in general, we are more capable of finding solutions of relatively low quality as opposed to solutions of high quality. One hypothesis about the resulting distribution is that it can be described by contour lines which are growth curves, in the manner shown on the chart above.

A good deal of work remains to be done to fully characterize a typical Cost, Quality space. However, it can be seen that knowledge about the space for any given Quality score formula can be important for determining how to implement the needed operational system. For example, if one knew nothing

about the density of solutions in a Cost, Quality space, one might spend a great deal of time searching for a solution where the density is too thin for it to be reasonable to expect to find one. In fact, one could characterize the whole process of designing and developing a new system as a process of exploring a Cost, Quality space in order to find a system solution that will be acceptable both in terms of Cost and Quality. When looked at in this manner, system development can be characterized as a process of successively drawing new solutions out of the space rather than a process of continuous modification of a given approach. For some purposes, the former conception can be quite useful. For example, it does not bind one to a fruitless attempt to modify a basic approach to system implementation which has no related representatives in the desirable area of the Cost, Quality space.

Desirability

It will usually happen that the customer for the output of an operational system will be willing to trade a certain amount of Cost for a certain amount of Quality. Thus, usually it will be possible to chart lines of equal desirability in the Cost, Quality space such that any two points on the same line are equally desirable in the eyes of the customer.

After Cost and Quality formulas have been developed with the concurrence of the customer, it will ordinarily be useful to go one step further and to obtain the customer's agreement on a formula which will enable the identification of equally desirable systems that differ in Cost and Quality and which will enable systems of different desirability to be rank-ordered. By this method, a basis is established for comparing system solutions which differ in both Cost and Quality without need for the customer to be present to concur immediately in every comparison.

"A" Score Formula

The system defined by a Quality score formula is "an empty box"; it is a symbolically defined system for which there is temporarily no real-world counterpart. Such a symbolically defined system has a direct relationship

only with its follow-on system.[1] However, once the symbolically defined system is implemented by means of any physical system, many other direct relationships are established. We will refer to these other directly related systems which come into consideration only after system means have been identified as adjacent systems. In using the term adjacent systems in this manner, we will expand its meaning beyond that given for it earlier when we discussed the concept of adjacent systems as it is employed in talking about a symbolic array of related systems. We now include not only systems which provide inputs to the operational system of concern, but also systems which contribute important conditions under which the operational system must perform and systems which are directly affected by spurious outputs of the operational system.

It should be noted that if one moves from one system solution to another, the Quality score formula does not change, whereas the set of adjacent systems does change. For example, let us consider a Quality score formula and a target Quality score which call for the delivery of public electrical power. If we implement the system by means of a plant fueled by coal, the relationship of the plant to public health as an adjacent system is quite different from the case in which we implement the system by means of hydro or nuclear power. In all cases, however, the Quality score formula is exactly the same.

A given system solution for an operational system is likely to affect its adjacent systems in many measurable ways. Some of its effects will be desirable; others will be undesirable. A system which is quite acceptable from the standpoint of its position in the Cost, Quality space may, in fact, be completely unacceptable because of its relationship to adjacent systems. Therefore, the relationship of any candidate system solution to its adjacent systems should be described in a manner that will permit assessment of the acceptability of the relationships on an absolute basis. There is also need to express the relationship of a candidate system solution to its adjacent systems so that the candidate solution may be compared with other candidate solutions.

---

[1] When a Quality score formula includes system input measurements there will also be a direct relationship to adjacent input systems.

The formula for expressing the relationship of a given system solution to its adjacent systems is called an "A" score formula. Whereas the Q score formula remains the same for all system solutions, the "A" score formula is unique for each system solution. The fact that the "A" score formula is unique makes it difficult to compare competing system solutions in this regard. It is therefore important that the "A" score formula call for the expression of the relationship of a system solution to its adjacent systems in a manner which permits decision making among alternatives. Usually this will mean that the relationship should not be expressed as a single number as in the case of the Quality score.

It is highly desirable that the "A" score formula and the Quality score formula be kept separate. To combine them is to obscure the clear-cut first-priority basis for comparing system solutions which is provided by the Quality score formula. When the "A" score formula is maintained as distinct, systems may be equated for desirability on the basis of their positions in the Cost, Quality space and may then be compared on the basis of their positive and negative (desirable and undesirable) relationships as expressed by their "A" scores. In this kind of situation, "A" scores must usually be expressed in different units for different system solutions, and human decision-making capabilities are most required.

Constraints

Constraints are deliberately placed limitations on the freedom of the designer to choose the means by which system functions will be implemented. Ordinarily the word constraint is not used to refer to indirect or unintentional restrictions on the freedom of the designer to choose means.

Constraints may derive from consideration of the relationship of a physical system to its environment. For example, society demands the protection of human life, and, as a result, restrictions are placed upon an aerospace system which have nothing to do with accomplishment of the system mission (its Quality score) but which have to do only with the protection of people in its environment who may be harmed by its exhausts, or who may be injured in the event of its failure. Constraints thus derive from anticipating undesirable

effects upon adjacent systems when a specific method of system implementation is considered.

Constraints may be stated in terms which proscribe certain means or in terms which prescribe certain means (as in the case of building codes). Constraints may also be stated in terms of measured outputs which must be maintained (as in the case of sewage plant specifications) such that the effect of the constraints upon means selection is to enforce the use of certain classes of means.

The term constraint can be preserved for the special purpose given here by employing restraint as the general purpose term.

Basic System Specification

In order to avoid misdirected effort and the attendant waste of resources, it is desirable to approach any development program with a complete description of the test of success that will be applied at the end of the program. If specification of the test which must be "passed" is not given at the beginning, then it is highly unlikely that all of the effort within the development program will be devoted toward the desired but unstated goal.

The term Basic System Specification refers to a document which identifies the manner in which the output of a system development program will be tested. The Basic System Specification serves as an "order" to initiate development of a specific system, it provides a single criterion for all development activities during the progress of the program, and it serves as the basis for a test of the output of the development program. In the last of these three roles, it also serves as a public statement by the customer of the test which the developed operational system must pass in order to be acceptable to him.

The Basic System Specification for any system would normally contain a Quality score formula and the target Quality score, the Cost formula and identification of any limits on resources to be used, a Desirability score formula, an "A" score formula for each method of system implementation to be considered with identification of any constraints resulting from consideration of specific adjacent systems, identification of the customer and the specific follow-on

to be served, and a statement of concurrence in the Basic System Specification by the customer. A document with this basic information will serve to identify the manner in which the end products of the development program will be evaluated and thus will serve as an order to initiate a development program directed toward a specific and publicly identified goal. The suggested content of a Basic System Specification given above is, of course, not intended to be all inclusive. What is required is that the Basic System Specification contain all of the information necessary to identify the manner in which the end product of development will be tested and judged, and that it have the public approval of the customer so that it may stand as a single criterion against which all system development activities may be evaluated throughout the course of a system development program. At the heart of the Basic System Specification is the Quality score formula and the target Quality score which provide a basic "one function" definition of the operational system that is required.

Aerospace System

An aerospace system is an operational system with remote and local segments. The remote and local segments are physical packages which are configured so that the local segment can move through space relative to the remote segment. Thus, the remote segment is exemplified by a launching platform or airfield, and the local segment is exemplified by a rocket or an aircraft. Although the remote and local segments are separate physical packages, they operate together as a system defined by a single Quality score formula. An aerospace system always includes a propulsion function, and when the local segment is manned it always includes a personnel support system (see the following discussion).

In general, aerospace systems are transportation systems; that is, the measure of Quality is in terms of success in transporting people or goods. However, aerospace systems frequently carry "piggyback" systems, and this is especially true in the case of space systems. A typical piggyback system is one whose output is scientific data. When an aerospace system and a piggyback system are joined together, there must be a unique Quality score formula for each system.

The remote segment of an aerospace system is always manned. The remote segment may be fixed on the Earth's surface or it may itself move through space. For example, an orbiting launch facility may be considered to be a remote segment which moves with respect to a point on the Earth.

Personnel Support Systems

Whenever we state that the reliability of a means for implementing a given function is r, it must be understood that any value given for r is valid only for a relatively restricted range of environmental conditions. Thus, if one or more of the environmental conditions under which the means will operate is very different from those assumed in giving a value to r, then the reliability of the means in the operational situation may be very much less than expected. For example, the reliability with which a block of ice will support a weight when the temperature is below 0^{o} Centrigrade may be quite different from the reliability with which the block of ice will support a weight when the temperature is above 0^{o} Centrigrade. It is apparently true that whenever we give a value for the reliability with which a given means will implement a specific function, we must append a statement of the ranges of important environmental conditions under which the stated value will be true. It will then be possible to predict the reliability of the means in the operational system only when it can safely be assumed that the stated conditions will obtain.

Many times in selecting means to implement component functions of a system, there is no option but to select means which will have the required reliability only under tightly controlled environmental circumstances. This is just as true of hardware as it is of personnel as means. Electronic devices require the control of temperature, humidity, simple life forms, and particulate contamination in the atmosphere just as human beings do. Even the simplest of hardware means require temperature control so that extremes are not exceeded. Because the reliability of each system means is apparently always related to the environmental conditions under which it must perform its assigned function, a system may require one or more "support systems" which are concerned with providing the environmental conditions necessary so that prime and additive means will perform their assigned functions with the required reliabilities.

One may ask whether a support system is prime—or whether it is an additive set. The answer is that it is neither. The concepts of additive loops and prime functions are independent of any need to consider means for implementing the functions. On the other hand, a support system has its effect upon means—specifically upon the reliabilities of means. To have any effect at all, the means upon which a support system acts must have a reliability greater than zero under some environmental conditions. If a means never has reliability greater than zero under any environmental conditions, then a support system which controls environmental conditions for it cannot have any effect upon the probability of success factors in the Quality score formula.

Under severe environmental conditions of use of an operational system, the failure of a support system may mimic the failure of a prime function. That is, the failure of a support system may reduce the reliability of one or more prime system means to zero and thus have the same effect as removal of the prime means. An obvious example of this kind of effect would be the failure of a support system to provide temperature control for an astronaut with resulting death of the astronaut and loss of his capability to implement prime system functions.

The effect of support systems is specific to overall system probability of success just as the effect of the additive set is specific to overall system probability of success. However, the additive set is articulated with the prime system by means of monitoring functions which respond to prime output failures. Unlike the additive set, support systems operate without monitoring system "throughout." A support system may support the reliabilities both of means which implement prime functions and of means which implement additive loops.

The concept of a personnel support system, that is a system which provides environmental conditions for men in an operational system, is important in aerospace system development. Men in the local segment of an aerospace system are virtually always exposed to environmental conditions which would degrade the reliability of performance of many functions if support systems were not provided. A support system for the local segment of an aerospace

system is called a Human Support System; for the remote segment it is called a Safety and Support System. Either of these personnel support systems includes all of the means justified for inclusion in the system on the basis of requirements for environmental conditions to sustain the reliabilities of prime and additive system means where the means is man.

Development Cycle

Let us consider a typical class of functions in which the input state to a function is a Primitive Need Statement and the output state is a delivered installed operational system with data to demonstrate that it solves the problem of its follow-on system in a manner that is acceptable to the customer. We will call any process that may be placed in correspondence with such a function, a development cycle. Thus, a development cycle is a discrete system which is initiated by a Primitive Need Statement and which delivers at a point in time the means which constitute an operational system. A development cycle is therefore a type of system which can be studied using the simple calculus for discrete systems described in Part B.

It is recognized that some processes which would normally be called aerospace system development cycles do not produce an output at a point in time and thus cannot strictly be described as discrete. However, by and large such aerospace system development cycles (for example a development program which produces many copies of an SST) may, with minor conceptual adjustments and allowances, be treated as discrete systems. Thus, one may temporarily think about such development processes as though they were intended to produce a single aircraft, rather than a number of them, and later make adjustments for the difference between the temporary conceptual approach and what is truly required as an output.

In this study we will assume that any real world aerospace system development cycle can be considered as a discrete system and that it can therefore be placed in correspondence with the development cycle model given in this report.

Performance Capability

When a means is selected by a designer to implement an operational system function, it is selected for its performance capability. That is, it is selected because it has a latent capability to satisfactorily implement a process which is symbolically described by a functional specification. The performance capability of a means may in many cases be demonstrable before installation and activation of the complete operational system; however, a means is selected for its latent capability to implement functions in the operational situation, not in prior tests. Means are evaluated by acts of measurement based upon functional specifications derived by partitioning of the system; to evaluate a means by the use of an arbitrarily selected act of measurement is not to determine its performance capability.

"Go" Model

No aerospace system development cycle could be likened to the course of an arrow to its target. Any real development cycle strays from its true course many times and must be brought back by corrective action. In one sense, the successfulness of a development cycle could be measured by taking a count of the errors made along the way which required expenditures of resources to correct, or which represented the useless expenditure of resources following pathways to dead ends.

One way to generate a model of a development cycle might be to partition a development cycle into component functions and then to attempt to show all of the adjustments that might be necessary for every bad intermediate state that might occur. The resulting model would include an horrendous amount of detail, most of it directed toward describing what <u>might</u> happen if an error were made in the process of system development. A model with this kind of detail is not readily accepted simply because it is so difficult to use. An alternative approach is to partition a development cycle without taking any account of the reliability with which each component function might be implemented, and thus without any consideration of possible additive loops and all

of the complexity attending them. Such a model which assumes that each function is implemented without error is referred to here as a "GO" model.

The basic development cycle model that is presented in this report is a "GO" model. There is a separate section of the report which concerns itself with principles which may be employed to elaborate the model in order to take account of errors in the course of system development. By this device, the model reveals the basic underlying structure of an aerospace system development cycle without unnecessary and occluding embellishment. Although at first glance the model may appear to be impractical because it ignores errors in system development, it is indeed much more practical as a device for assisting in the prediction and control of aerospace system development than a model at the same level of detail which considers all possible development errors and functions for their adjustment.

Reliability

When we talk about a function, we refer to the probability of its output. We implement a function defined in this manner with means, and we then talk about the reliability of the means. This usage can be misleading. More properly we should refer to the reliability of means performance. The idea of reliability of a means per se is not a useful one. Any given means (object) may have many different reliabilities depending upon what function it is employed to implement. Thus, for example, a block of cement which is used as a counterweight in one system may have a very different reliability for that function as compared with the case in which an equivalent block of cement is used in another system under compression to support a mass.

It follows that whenever we speak of the reliability of a means, care must be taken to identify the function which it is intended to implement with the stated reliability.

The reliability with which a means implements a given function will depend upon the environment in which the means is to be employed. Therefore, the concept of reliability of a means also carries with it an implication that the

environmental conditions under which the stated reliability will be observed must also be specified. (See the discussion of personnel support systems in this section.)

Reliability is thus an attribute of a means when the means is employed to perform a given function. In design, one problem is to find means with reliabilities which match probability of output requirements defined in functional designs.

Development Quality (Dev Q)

If we accept that an aerospace system development program may be treated as a system, then the concept of a Quality score may be applied to a development cycle in a manner similar to the way in which it is applied to define an operational system. We will refer to any formula for measuring the quality of a development cycle as a development cycle quality formula, or simply as Dev. Q.

Ideally we should hold that an operational system is not properly defined as a system until a Quality score formula has been approved. The generation of the Quality score formula must be accounted for within a development cycle. The initiating input of a development cycle is a Primitive Need Statement, not a Dev Q formula. This means that the formula by which the success of a development cycle itself will be measured must be generated within the development cycle. The development cycle model presented in this report provides for the generation of such a formula. Two important factors in this formula are the probability of success of the development cycle (prediction of success) and the calendar time at which the development cycle must produce its output. The concept of development quality is employed primarily for the purpose of predicting the quality of alternative development cycles rather than after the fact as a measure of success.

Management

The concept of a Quality score for a development cycle calls attention to the fact that a "good" development cycle is not simply one which delivers

a satisfactory operational system. The operational system must be delivered on time and within the resources allocated for the development cycle. However, the basic "GO" model presented in this report is designed to account only for the delivery of a satisfactory operational system as the output of an aerospace system development cycle. [1] It does not include identification of those functions necessary to account for delivery of its end product on time and within cost.

To provide for a high quality development cycle per se, a number of general management functions must be provisioned. By general management functions we mean functions within the system development cycle which are justified because their effect is to ensure that the development cycle will itself be of high quality. Typical general management functions are those which:

1. Provide for the design of the development cycle itself;

2. Provide for the effective use of the resources necessary to carry out development;

3. Provide for the detection of deviation from time and cost targets;

4. Provide plans and means for correcting a development cycle which has a time or cost error;

5. Provide for liaison between the customer and the development program and between the funding agent and the development program.

Consideration of the way in which management may be taken into account in using the basic development cycle model presented in this report is discussed in Chapter V of this report.

---

[1] The model therefore accounts for what might be called "technical management." By technical management is meant the management necessary within the development cycle to assure the quality of the delivered end product as measured by its own Quality score formula.

Allocation of Function

In the design of any system, functions must be specified before means can be selected to implement them. Without prior functional specification, there is simply no criterion for the selection of means. [1] Inasmuch as function specification must precede, the task of the designer is to consider alternative means by which a specified function may be implemented and to assign a specific means to the function. This process may be called means allocation.

In some cases, the process of means allocation will result in the determination that the system under design will include a general purpose means such as a man, a computer, or a power supply. Whenever a general purpose means is called out and is justified in terms of its effect upon overall system Cost and Quality, then it is usually highly desirable to take full advantage of the general purpose means by loading it to capacity — that is, by assigning to it responsibility for carrying out functions other than those which gave rise to its selection. This process of identifying all the functions which should be assigned to an already justified general purpose means because of the Cost and/or Quality benefits which accrue may be called function allocation. Function allocation is thus a process which is in a sense the inverse of means allocation; it is employed only when a general purpose means has been justified for inclusion in the system under development.

Personnel Products Package

Our stated purpose in presenting a development cycle model is to enable predictions about the course of development of man-related products in aerospace system development cycles. In order to provide for the man-related

---

[1] Despite this fact, it frequently appears that means are selected in system design without prior functional design. It can certainly be shown in every case in which this appears to be the fact that means have been selected against implicit rather than explicit functional specifications. (See also pg. 64.)

content in the full model, it was necessary to identify the man-related end products which must be accounted for by the model. Thus, in developing the model it was necessary to put detail in the output description of an aerospace system development cycle such that the partitioning of the development cycle to produce the model would be forced to account for all of the man-related end products. The end product listing that is presented below was derived by logical analysis and by the consideration of typical aerospace systems (see Appendix A). Inasmuch as the end products of an aerospace system development cycle are the means of the operational system, we will list the man-related end products as means or things. Five categories are sufficient to encompass the end products:

1. Selected and trained crew members.

2. Job aids.

3. Materials to maintain reliability of crew performance on the job.

4. Products of human engineering including interface devices, tools and workspace arrangements.

5. Personnel support systems (Human Support System and Safety and Support System).

Taken together, we will refer to all of these end products as a personnel products package, or simply as personnel products. All of these classes of things are installed as physical components in the operational system. All of them relate to man as an implementor of system functions — both prime functions and additive functions. All common man-related end products of aerospace system development cycles fit into one of the five categories.

Selected and trained crew members. — Properly selected and trained crew members will bring to the operational system capability to perform specified operator and maintenance technician performances[1] which have

[1] Human performance which implements a prime system function is called operator performance in these reports. Human performance which implements

been identified in the final system design as functions to be carried out by man in the system. The use of man in this manner is analogous to the use of hardware to implement system functions.

Job aids. — Sometimes, capability to perform operator and maintenance technician functions will not be in the learned repertoire of crew members but will be supported in whole or in part by job aids. Thus, the means for implementing some functions will not be man alone but will be a man plus a job aid which is specific to the function to be implemented. Job aids are thus part of the means package necessary to obtain some operator and maintenance technician performances.

Materials to maintain crew member reliability. — In the two categories just discussed above, we have been concerned with man and man-plus-job-aids as means with inherent reliability for performance of each assigned function. However, the inherent reliability with which a man can perform an assigned function is not always good enough. When the reliability that is demanded exceeds what man can deliver on the basis of inherent reliability, many times it is possible to achieve the target probability of success by providing an additive loop to the man performance. Such additive loops are implemented not only by human performance but also by things which must appear as end products of the development cycle. These things, such as practice materials, simulators, and self tests, must be delivered as part of the operational system when it is installed. These things taken together with the man capability to employ them constitute additive loops of the first-order and lower order levels. The role of these additive loops within the system is the same as the role of additive loops which back up hardware performance and they relate to system quality in the same manner.

---

an additive function is called maintenance technician performance. The term "operator" is not used inasmuch as it implies a crew member who is assigned only operator performance — an option that is seldom justifiable.

Products of human engineering. — We do not refer here to all of the products which are sometimes associated with human engineering. Rather, we refer specifically to the human engineering designs which configure man-machine interfaces and man-man interfaces. These immediate outputs of human engineering efforts never appear as end products of a development cycle; however, the ultimate configuration of the hardware which is delivered and installed as part of the operational system embodies these human engineering designs. These man-machine interface "products" are directed toward achieving the reliable articulation of men and machines so that total system probability of success will not suffer.

Support systems. — As we have shown earlier, support systems for man in an acrospace system implement functions which are neither additive nor prime; the relationship of a suport system to overall system quality is determined by its factorial effect upon the reliabilities of each additive and prime means in the system. The personnel support systems for crew members are thus required for their effect in sustaining the reliability of each element of human performance in the system.

III. THE INDEX MODEL

There are over one hundred component functions in the development cycle model presented in this report. The shortest path through this model is over fifty functions in length. Ordinarily when one is using such a model for a practical purpose, attention is focused on a relatively small portion of the total, and it is impractical to employ a symbolic representation of the entire model which shows detail where it is not needed. To make it possible to have detail in the area of concern to the user of the model, and, at the same time to provide for an overview of the remainder of the model, an eight-function index model has been prepared. Each function in this model is indexed to a detailed description of its component functions. One may thus employ the breakdown of any single function of the index model for detail while using the index model itself to obtain context.

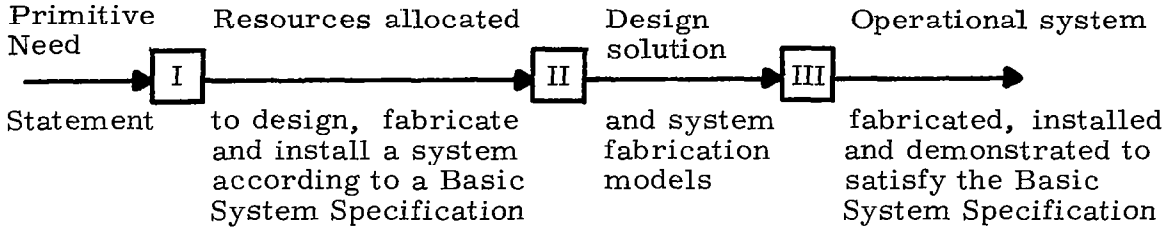
The index model is presented in a manner designed to assist the reader to memorize the important features of it. To foster memorization, we present a rationale to justify its form as well as a simple symbolic representation of it. Not only will the rationale aid the reader in memorizing the important features of the index model, but it will also help him to gain confidence that the index model is a fair and useful representation of an aerospace system development cycle at the "eight-box" level of detail. Such confidence is necessary if the index model is to be accepted as a tool for providing context.

Inasmuch as the index model contains but eight functions in series, it is simple to present a symbolic representation that can readily be examined. However, it is difficult to present the rationale for the index model all at once; it is easier to present it in two stages.[1]

---

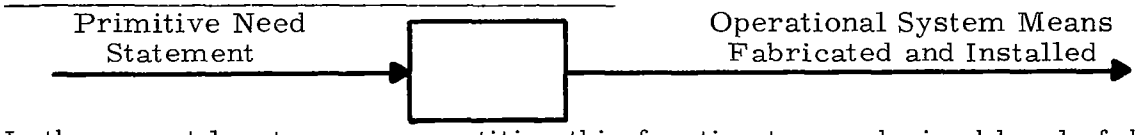
[1] A detour to examine the reason for developing the rationale by stages will be useful, because throughout this report the rationale for the model will be developed by working in stages from more comprehensive to less comprehensive levels of concern. The complete development cycle model has been developed by partitioning the generalized function description of a development cycle:

First, the single-function definition of a development cycle will be partitioned into a three-function model which we will call a phase model. Then each function in the phase model will be partitioned in turn to yield the eight-function index model. Let us therefore present the phase model and turn immediately to the rationale underlying it. In the future, references to Phases I, II and III will be to the functions defined in the symbolic model below.



Phase I Rationale

The input to Phase I is the Primitive Need Statement which initiates the development cycle as a whole. The output includes the allocation of resources necessary to carry out Phases II and III (design, fabrication, and installation of the operational system). Thus, the output of Phase I is marked by the decision to commit the funding required for the complete design and fabrication of



In theory, at least, we can partition this function to any desired level of detail. In practice, this must be done by a sequential partitioning process in which each partitioning generates increased detail. For each level of detail that is generated in this manner there must be consideration of what is the most important partitioning to make. Thus, if the function is to be partitioned into only two component functions, then there must be consideration to justify the specific partitioning that is made as being one which presents the most important information for the purpose at hand. The rationale for a "100-box" model was therefore generated in sequence, level-by-level, with the rationale for less detailed levels taking precedence over the rationale for partitioning at more detailed levels, the latter being possible only within the boundaries set down by the former. The rationale that will be presented in this report is therefore more easily presented by levels.

an aerospace system. In comparison to the funding committed at this time, the resources expended in prosecuting Phase I usually will have been quite minor. In conjunction with the dollar commitment, there will be a Basic System Specification which is a requirements statement for Phases II and III taken together. The Basic System Specification will also incorporate the data employed as a basis for the decision to commit resources. The facts presented in the Basic System Specification will include a Quality score formula for the desired system (incorporating an "A" score formula) and an identified target Cost, Quality position for the system to be developed.[1] Commitment of dollar resources normally requires that data have been developed to define the measure of quality of Phases II and III (Dev Q) and that the prediction of success for the development cycle is sufficiently high to warrant support. Therefore, the output of Phase I will also include a Dev Q formula and a justified prediction of success.

The test of goodness of the output of Phase I which would be employed by an all-knowing being would tell whether or not a correct decision had been reached with respect to the commitment of resources, and would tell whether the data included in the Basic System Specification were good and complete. As a practical matter, it is difficult, if not impossible, to obtain a satisfactory test of the output of the Phase I effort.

Is it necessary to pass through the output state of Phase I between initiation and completion of an aerospace system development cycle? From the standpoint of funding, it is clear that there are strategies which are alternative to the one implied here. For example, funds may be committed incrementally from the beginning to the end of the development cycle. By and large, this strategy is in ill repute, for it increases the likelihood that a relatively large expenditure will be made prior to cancellation of a development cycle, but it is a possible strategy and it must be recognized that the commitment of

---

[1] The Cost, Quality position may be presented as the lower boundary of Desirability that will be accepted for the product of Phases II and III. In any case, it will have been shown in Phase I to be an achievable position and to be one that is not unduly permissive in view of what the state of the art can be expected to deliver.

total funding is not a necessary element of the output state for Phase I. On the other hand, the Basic System Specification which also appears as an element of the defined output state is essentially a "one-function" definition of the system to be designed and to be fabricated. It is necessary that such a definition be made before design begins.[1] It may be made poorly, but we are not concerned at the moment with its quality. A definition must be given in some form; else, design has no goal. Therefore we may say that Phase I as demarcated on the output side by a formulated Basic System Specification is a necessary component function of a development cycle and that it must precede design and fabrication.

The output state of Phase I marks a major discontinuity in the development process. Thus, prior to the output of Phase I, there is no confirmed objective definition of the operational system to be fabricated. After Phase I, as defined here, the Basic System Specification provides a constant external criterion for evaluating all of the design and fabrication activities of Phases II and III, and for evaluating the end product of Phase III. In Phase I, activity may be characterized as a search for a definition of a system that is to be built. After the completion of Phase I, activities in Phases II and III may be characterized as a search, first for a design solution and then for a real-world implementation of the system defined in the output state of Phase I.

Within Phase I the customer plays an interactive role. Thus, whoever implements Phase I acts in a sense as technical advisor to the customer, assisting him to select a Basic System Specification that will call for a system to serve his needs, and assisting by presenting to him the data required to enable a proper decision whether or not to commit resources to development. Throughout Phase I there is a sequence of appeals to the customer to help structure the Basic System Specification according to his needs. After Phase

[1] The output of Phase III is a means, a complete operational system. To carry out the design and fabrication of this means in Phases II and III, there must be a criterion for accepting or rejecting candidate means. The criteria are necessary and must precede means selection; functional design (the specification of criteria) must precede means design. The necessity and priority of functional specification is discussed again later in this section under the heading, The Partitioning of Phase II into Component Functions.

I, the Basic System Specification stands as the word of the customer and provides a constant criterion for activities in Phases II and III.

To carry out Phase I requires a strategy that will yield a proper decision at small cost. Trial design and fabrication of operational design and fabrication of operational system components is not excluded from Phase I, but when possible Phase I will be carried out as a paper exercise so that expenditures will not become excessive. The goal of Phase I is not to restrict unduly the freedom of choice of designers in Phase II; more properly it is to show that the design problem of Phase II can be solved within the state of the art with a high degree of confidence. Thus, any design work undertaken in Phase I is exemplary rather than directive with regard to Phase II.

Phase II Rationale

The input to Phase II is the Basic System Specification and the associated funding support which is specified as the output of Phase I. The key output includes all of the models necessary to enable fabrication of the operational system and all of the associated fabrication tools that will be required. On the hardware side, models to enable fabrication will include, for examples, production drawings and specifications for the procurement of off-the-shelf items. On the personnel products side, the models will include training programs and training equipment, for examples. (Training programs are analagous to production drawings, and training materials are fabrication tools.)

It must be kept in mind that the model we are discussing has been simplified from a complete prescriptive model in several ways. One of the simplifications is to retain in the model the necessary order of events, assuming that the development cycle is "Go" all the way. Therefore, the line of demarcation between Phases II and III appears to be quite simple in the model. In fact, in the real world of system development the line of demarcation will be a shaggy one. Thus, some fabrication models will be completed before others, and it will be possible, or necessary, to begin fabrication of some items before others. The function model that we present here is not intended to deny this state of affairs; it is simply an inappropriate model for showing

such time relationships. The time relationships that must be set up to take into account retrofit time, and to ensure consideration of necessary lead time, are better shown by a different kind of model—a PERT model for example.

Given our present capability in the realm of system development, it does not appear possible to change the relationships among Phases I, II, and III. The development of instructions for fabrication of an operational system must await a definition of the system that is required, and it must precede the actual fabrication which it is intended to guide.

The output state of Phase II is a necessary one. It is not within our capability successfully to fabricate a complex aerospace system without first having described in detail its components and their relationships. Nor can we fabricate complex systems without fabrication models.

Within Phase II, development proceeds from the gross overall model of the operational system which is given in the Basic System Specification through successive stages of elaboration to the very detailed description of the operational system means that is contained in the set of fabrication models. In the main, this process is carried out on paper. The on-paper designs may be accompanied by the development of mock-ups for verifying expected relationships and by the development and test of prototype equipments. But such ventures into hardware are supportive to the development of the symbolic models; they are not in the main line of activity in Phase II. Thus, the objective of Phase II can be characterized as one of finding a paper model which can be set in correspondence with the Basic System Specification on the one hand, and in correspondence with a real-world implementation of the system on the other.

Phase III Rationale

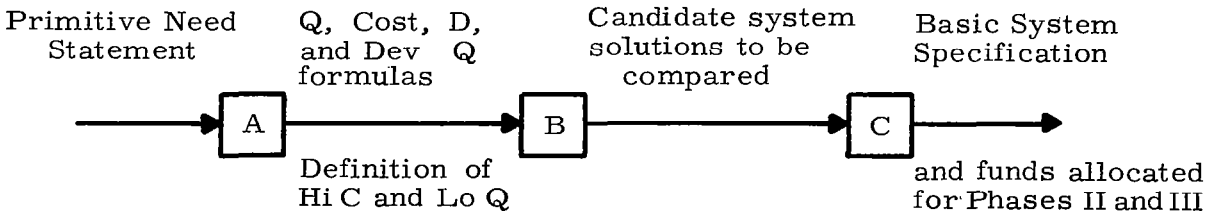
The initiating input for Phase III is, of course, the output of Phase II, the set of fabrication models. The output of Phase III is the output of the development cycle itself, the fabricated, installed operational system means, and the data which demonstrate that the installed system is capable of satisfying the Basic System Specification.

The output state of Phase III is a necessary terminal output by definition of the development cycle. Its position relative to Phases II and III is not open to question.

Within Phase III the process of development is in one sense opposite in direction to that of Phase II. Whereas Phase II proceeds from a gross characterization to a very detailed set of models, Phase III proceeds from the fabrication of detailed components toward a unitary assemblage which satisfies the original gross characterization of the system as a single function. The fabrication processes in Phase II include not only the manufacture of hardware, but also the training of system personnel, such that the delivered and installed product will embody all of the elements of performance capability necessary to implement the operational system.

The Partitioning of Phase I into Component Functions A, B, and C

The first three functions of the index model are shown symbolically in the figure below: they are the component functions of Phase I. Therefore, the input to the first and the output of the third correspond to the input and output states of the Phase I function.



The objective of Phase I is a key to the partitioning of Phase I. Let us, therefore, reconsider the requirement for the Phase I output. This will preclude attributing to Function C purposes which are commonly inappropriately associated with it.

What is required at the completion of Phase I is that the resources "power" needed for Phases II and III be allocated, and that the criteria be set down

which may be used throughout Phases II and III to guide and test progress. In principle, to guide the progress of design and fabrication and to test the adequacy of the output of Phase III does not require any consideration of the "inner workings" of the physical system that is developed. The guidance objectives may be met best by describing a test of the system that is to be developed in terms of the effects it must have when it is operational without direct consideration of how it is to be implemented. Therefore, it is not necessary that the output of Phase I concern itself with the means by which the system will be implemented in Phases II and III. In fact, it is desirable that the Basic System Specification say as little as possible about means because to do so is to limit the freedom of the designer in Phase II on the basis of less adequate information than will be available to the designer after Phase II has been initiated. We may therefore conclude that the Basic System Specification must set down an adequate test by which the system to be developed may be evaluated, but that it should not place constraints upon the design except when it is absolutely necessary and justifiable to do so. When we consider Function C in the discussion below, it will be seen that it includes consideration of means by which the needed system may be implemented. The purpose of such consideration is to determine whether or not a system can be built that will satisfy the need for it, how good the system can be, and how little it can cost. Thus, the consideration of means within Phase I will be for the purpose of developing the data necessary to justify a recommendation to proceed with Phases II and III; its major purpose will not be to determine how to implement the needed system.

Let us now consider the cut which establishes the output state that separates Functions A and B. This cut demarcates consideration of "exterior design" on the left and consideration of "interior design" in Function B on the right. Function A on the left is focused exclusively on determining all of the attributes of the needed system that derive from consideration of its follow-on system and the customer.[1] The categories of information that may be

[1] Throughout we will consider the customer and the funding agency to be the same unless otherwise noted.

developed in this manner include the Quality score formula, the Costing formula, the formula for estimating the quality of the development cycle that is anticipated (Dev Q), and, if required, the formula for computing Desirability. If a "D" formula is not given, the output should at least include a definition of what the customer considers to be the lower bound of Quality and the upper bound of Cost that will be acceptable. Taken together, these elements of the output state of Function A bound the system to be developed in such a way that they provide for a test of any developed system in terms that will permit identification of its effects upon the follow-on system. They provide a top-level specification in boundary terms within which there may be an exploration of alternative ways by which to implement the needed system and thus to satisfy the customer. The output of Function A is therefore a needed input to Function B where identification of alternative methods of implementation must be made. Function B, in a sense, then generates the grist for the mill which is Function C. Thus, the output of Function C (the output of Phase I) cannot be achieved without consideration of alternative methods of system implementation. Before alternative methods can be considered, they must be identified. The identification of those to be considered in Function C is the output of Function B. The criteria by which candidate methods of implementation can be recognized are contained in the input to Function B.

Function A

The input to this function is the Primitive Need Statement which initiates a new development cycle. The output state includes the formulas listed above, a definition of the Cost and Quality boundaries within which an acceptable system solution must fall, and a Need Satisfaction Score formula. The Need Satisfaction Score is an intermediate product within Function A which will be given detailed consideration later in this report when the rationale for the partitioning of Function A is presented. It is useful, however, to consider this interim product as one which is carried forward to appear as an element of the output state of Function A.

There is no question that the output state of Function A can precede the output states of all subsequent functions in the index model. All subsequent

functions require consideration of internal design; Function A requires only information about the follow-on system and the customer, which presumably is available at any time. The relevant question with respect to the ordinal position of Function A is whether or not it must come first. Inasmuch as the output state deals with the objective specification of what is needed and relates that specification to need satisfaction, there appears to be no alternative but to essay Function A in some form before it is possible to begin consideration of the interior design of the operational system. Without the output state of Function A, there is no defensible way to state the objectives of the subsequent development functions nor to evaluate their end products. Thus, Function A must precede all of the rest, and its output state is a necessary one within the development cycle.

Because it may appear that Function A frequently is not implemented in practice as the first step of development cycles, there may be some concern about the assertion that it must come first. In truth, it probably can be shown that Function A has been the first function of every aerospace system development cycle. However, the function is frequently carried out badly. That is to say, often a very rough approximation of the output of Function A is accepted as an "understanding" and is employed as a basis for undertaking Function B. When this is done, an explicit specification is sometimes developed later as an output of a subsequent function, but many times it is simply never expressed. When it is not expressed, work within the development cycle usually proceeds on the basis of an unspoken understanding among participants. Unfortunately, unspoken understandings frequently turn out not to be understood at all.

In the case of a real development cycle, in order to complete the work of Function A there will need to be constant appeal to the customer to obtain his agreement with respect to the various formulas to be developed, and to obtain his definition of unacceptably high costs and unacceptably low quality. If we think of the customer as being outside of the development cycle, then Function A is primarily an activity in technical support to the customer; it elicits from the customer needed information and puts it in a form that will be useful in providing guidance for the remainder of the development cycle.

Function B

The initiating input of Function B is the output of Function A. Its output is a list of candidate system solutions selected on the basis of their estimated Cost and Quality and probability of success of development. The system solutions identified in the output cannot, however, be certified to be certain of success of development, for to establish certainty it would be necessary (within Function B) to carry out the full development program for each candidate system solution. It can be said, however, that the candidate system solution list identified in the output of System B will not contain a large number of methods of system implementation that are clearly unfeasible.

The output of Function B must precede the output state of Function C because the data contained in the output of Function C relate to a subset of system solutions drawn from the set of system solutions identified in the output of Function B. The set must be identified first.

In performing Function B, it is useful to "test" suggested methods of system implementation in order to reject suggested solutions which cannot be justified as candidates. Thus, in a sense, Function B embodies a "filtering" action. If the output of Function B contains a high proportion of system solutions that are not bona fide candidates, then it will "overload" Function C.

Function C

This function is initiated by the output of Function B; its output is the output of Phase I, an objective requirement statement for Phases II and III and the resources necessary to prosecute these phases. Inasmuch as the assignment of resources should be contingent upon a demonstration that Phases II and III will produce a system of given Cost and Quality with accepted prediction of success, information characterizing the expected Cost and Quality of the system to be developed will be an element of the requirement statement.

We have already shown that Function C must follow Function B within Phase I. The necessity of its output statement has previously been demonstrated in developing the rationale for Phase I.

To perform Function C requires consideration of alternative ways within the state of the art to design and fabricate the required operational system. The purpose of this investigation, however, is not to set forth a design, but rather to test whether or not an acceptable design is possible. If an acceptable design is possible, then the purpose is further to determine the best Cost, Quality position that might be achieved by carrying out Phases II and III. Such estimations can be made on the basis of representative operational system designs and plans for fabrication; firm plans need not be made. To find plans for representative systems in the most desirable Cost, Quality area requires consideration of all of the applicable state of the art. This will be an overwhelming task unless a strategy is found to reduce the number of alternative combinations of candidate means approaches and candidate functional approaches to be considered. The needed strategy will be considered in the rationale for the partitioning of Function C.

The Partitioning of Phase II into Component Functions D, E, F, and G

Phase II may be characterized as a process that is directed toward finding a means description which can be put agreeably in correspondence with the description of the required operational system given in the Basic System Specification. To be in correspondence, the means design must imply a real-world system whose Cost and Quality will fall within the target Cost and Quality area. Thus, the means design is not a certain one, but is any description from the total group of such descriptions in a specific area within the Cost, Quality space. The means design that is the output of Phase II must be in the form of a requirement for the Phase III output, and it must contain sufficient information to enable fabricating that output, the physical means to implement the operational system.

The problem to be solved in Phase II has two principal characteristics: 1) the number of system solutions in the Cost, Quality space associated with any given Basic System Specification may be very, very large indeed, and

2) design must approach the identification of the system solution in a stepwise manner that does not permit full confidence in any single step until after all of the steps have been completed. Thus, in Phase II the designer must start from the Basic System Specification and find a path through an ever-broadening maze of possible design alternatives, step by step, until he finally achieves a satisfactory system solution. What is required is a way of assuring that a successful path will be found and that it will be done with efficiency. The partitioning of Phase II into component Functions D, E, F, and G in the index model is based upon a strategy which satisfies that requirement. Before discussing the strategy, however, we will explore more fully the two-part problem situation identified above.

To provide an intuitive appreciation for the first part of the Phase II problem, the number of state-of-the-art system solutions that may be found in a Cost, Quality space, consider a Basic System Specification which calls for a man to be in New York City at a point in time, t_1, and to be in Boston at t_2, where t_2 is later than t_1, but not more than 24 hours later. Although it would not be useful to present the complete hypothetical basic specification here, let us assume that the only acceptable solutions are means which already exist. First, we should note that there are several classes of means that might be employed, such as private automobiles, busses, trains, and aircraft. Within each class, there are many subclasses. For example, within the private automobile class there is a matrix composed of automobile type by automobile manufacturer by year. Within each cell in this matrix, there are still many more means. Thus, a two-door Ford Galaxy, 1966 model, may come in different colors with different tires, different carburetors, different sparkplugs, and so on in many different combinations. In fact, the total number of different solutions available to satisfy the hypothetical Basic System Specification is very, very large indeed. If the Basic System Specification identifies a target Cost, Quality area, then many of the solutions in the Cost, Quality space may be rejected because they fall outside of the target area. [1] However, it is most likely that there will also be many remaining

[1] Suggested solutions may also be rejected for other reasons. For example, Dev Q or probability of success of development may be unacceptably low.

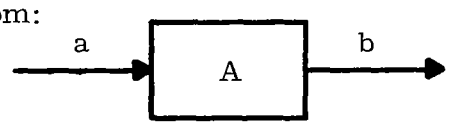
within it, and it is quite certain that one cannot know where a particular solution falls in the space simply on the basis of its name. Thus, to reject possible solutions requires sufficient study to characterize them in terms of their estimated Cost, Quality positions.

When a designer begins Phase II he must recognize that there is probably a very large number of solutions in his Cost, Quality space. Some will be acceptable; others will not. It will be his task to find just <u>one</u> acceptable one.

The second aspect of the Phase II problem is more difficult to characterize than the "number of solutions" aspect. However, it is important to appreciate that the achievement of confidence in each design step in Phase II is a problem of sufficient importance to be a deciding factor in the generation of the development cycle model. Therefore, we will turn next to an attempt to characterize this second part of the problem.

We have said that the designer starts Phase II with a "one-function" definition of the system that he must implement. The Basic System Specification which contains the "one-function" definition implies a unique Cost, Quality space which is populated with a very large number of system solutions. Let us signify the one-function definition by a simple one-function diagram, and let us symbolize the large number of system solutions in the Cost, Quality space by a string of Ss. (It will be understood that the number of Ss in the string falls far short of the number of system solutions in a hypothetical Cost, Quality space.)

Design proceeds from:



To one S from the set:

$S_1\ S_2\ S_3\ S_4\ S_5\ S_6\ S_7\ S_8\ S_9\ S_{10}\ S_{11}\ S_{12}\ S_{13}\ S_{14}\ S_{15}\ S_{16}\ S_{17} \cdots S_n$

Now each complete system solution symbolized by an S contains many, many elements of information. (Presumably it requires no demonstration to show that a very large number of pieces of information must be generated to enable the fabrication of an aerospace system.) The amount of information required to completely specify any S is usually too great for a designer to move in one step from Function A to any specific S. It can be seen then that the

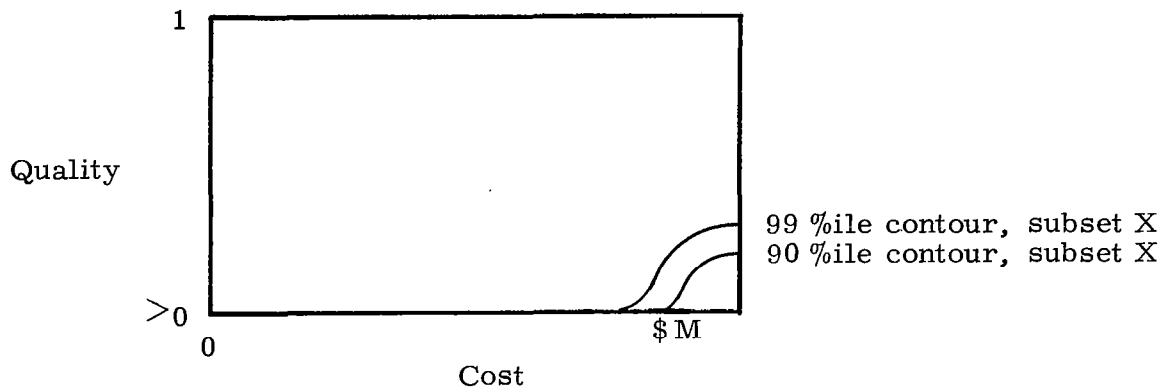
path from Function A to a specific S must proceed in a stepwise manner, and that each step must accumulate additional information about the S that will be the eventual endpoint of the stepwise movements. Now if there is a necessary order in which the steps of a stepwise approach must be carried out, then it is important to know that order, for the path from Function A to an S is unlikely to be an efficient one if the necessary order of steps is not known. Analysis reveals that there is a necessary order of steps such that violation of the order will introduce unnecessary and wasteful steps in the design process within Phase II. The outputs of Functions D, E, F, and G in the index model identify an inviolable sequence of steps which must be taken when Phase II is carried out in a stepwise manner.

Before presenting the rationale for the order of steps within Phase II, it is important to consider another aspect of the stepwise approach. The problem is simply this, "When a designer takes a step which moves him from Function A toward an S, how does he know that he is moving toward an S that is in the target area of the Cost, Quality space?" Examination of the string of Ss which represents the multitude of system solutions in the Cost, Quality space shows that some of the Ss will fall in the target area, but that most of them will be unacceptable, either from the standpoint of Quality, or Cost, or both. In order to determine the Cost, Quality position of a given S with complete confidence requires that all of the information content of S be available. However, when a designer takes a step toward an S, by definition he does not obtain all of the information necessary to completely specify it. Therefore, he may or may not be moving toward an S that is in the target area. What the designer requires is a way of approaching an S that will permit him to maintain confidence that each step he takes is moving him toward an acceptable S. Let us now consider a device by which a designer may obtain such confidence and by which he may therefore keep himself on a path from Function A that will move unerringly toward an acceptable S.

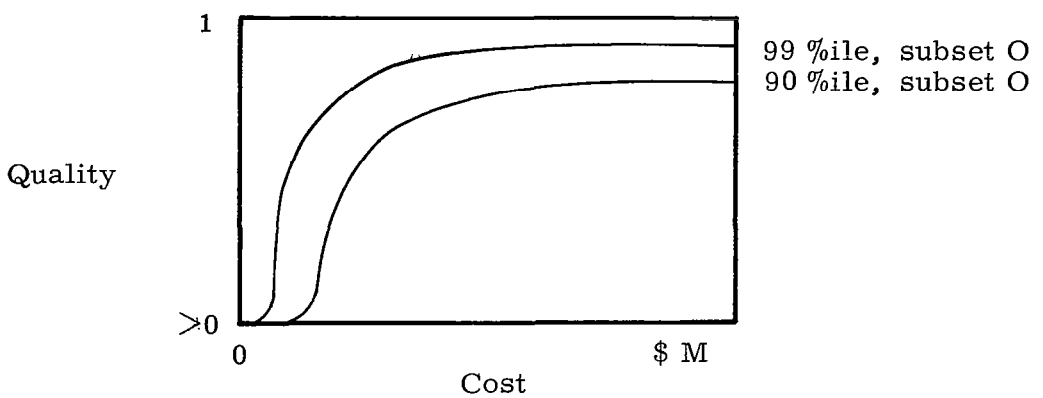
Let us consider again our string of S_S and let us without bias pick two subsets, denoting one as the X subset, and the other as the O subset.

$(S_1)\ S_2\ S_3\ \not{S_4}\ (S_5)\ \not{S_6}\ (S_7)\ (S_8)\ S_9\ \not{S_{10}}\ S_{11}\ S_{12}\ (S_{13})\ S_{14}\ \not{S_{15}}\ \not{S_{16}}\ S_{17} \cdots S_n$

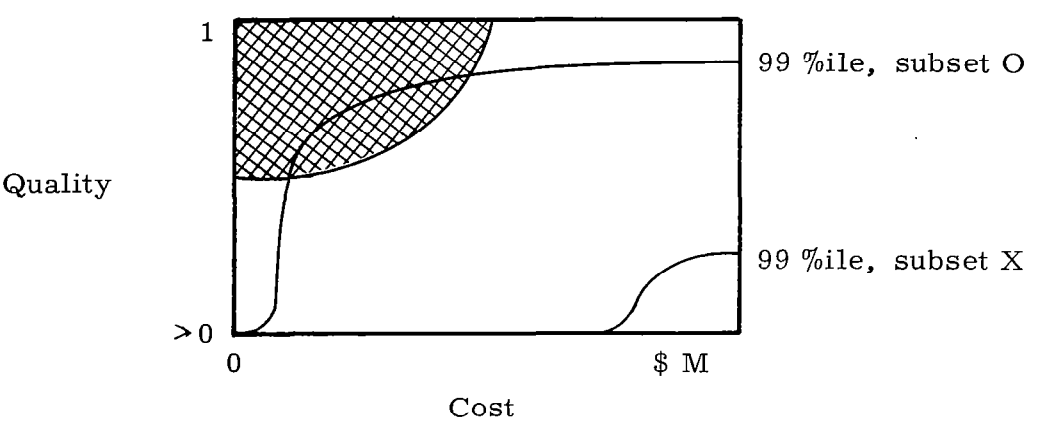
Any subset that we might pick will have its own distribution in the Cost, Quality space, and from subset to subset the distributions will differ. For example, the distribution of subset X might look like this:



and the distribution of subset O might look like this.



Now let us assume that our target Cost, Quality area is the shaded area shown in the diagram below.



It is quite clear that it will be very difficult to find a representative from subset X in the target area, whereas it should be possible to find a representative from subset O in the target area. From these examples we learn that it will be desirable to avoid taking any steps in Phase II which result in the selection of a subset with virtually no representatives in the target Cost, Quality area.

Each step that must be taken within Phase II will result in the identification of some information about the target S. Thus, each step will encompass a subset of Ss; it will encompass all of the Ss which are compatible with the information accumulated in the stepwise manner up to the point in question. What is desirable, then, is that information be provided which shows that the subset of Ss to which the accumulated information restricts future steps is a subset which includes Ss in the target Cost, Quality area. At the very least it is desirable that it be shown there is one S in the target Cost, Quality area; it would be most desirable if it could be shown that there are many Ss in the target Cost, Quality area.

Supporting and Stabilized Data

Now we can see that the output state of each step must encompass two things: (1) a statement of accumulated information which identifies a subset of Ss, and (2) data which show that the subset of Ss identified by the accumulated information includes one or more Ss in the target Cost, Quality area. We will talk about the first kind of information as stabilized design decisions. We will talk about the second kind of information as supporting data—data which demonstrate that subsequent steps toward S can be carried out in such a manner as to lead to an S in the target Cost, Quality area. The first category of information is called stabilized because it is assumed that it represents information accumulated in a stepwise manner that will not be changed. The second kind of information is called supporting data because it is presented solely for the purpose of providing confidence that the information generated in the stepwise manner can be stabilized without losing an opportunity to find an acceptable solution. The supporting data are not stabilized. The supporting data may be design data just as the stabilized data are design data. The

stabilized data document the path which has been taken from Function A toward an S with the understanding that it will not be retraced. The supporting data predict a path that could be taken in order to get from a present position to an S. The supporting data simply indicate that there is at least one such path. Thus, the supporting data do not restrict the future steps in any way; future steps which lead to an acceptable S may be taken without regard to the supporting data.

The strategy reflected in the part of the index model which corresponds to Phase II calls out a stepwise approach through Functions D, E, F, and G. It calls for the production of two kinds of data at each step along the way: (1) stabilized design data, and (2) supporting data.

Step Sequence Rationale

One more task remains before presenting the definition of the component functions of Phase II in the index model. The rationale underlying the sequence of steps which is said to be "necessary" must be presented.

Our first argument is that the design of the additive set cannot be completed before completing design of the prime system. Every element in the additive set (that is, every additive loop) must contribute to overall system probability of success. Conversely, an additive loop which does not contribute to overall probability of system success cannot be justified. The role of an additive loop is to make up for the difference between the required probability of output of a prime function and the reliability of the means by which it is implemented. Therefore, an additive loop cannot be designed prior to the identification of the prime function and the basic means by which the prime function will be implemented.

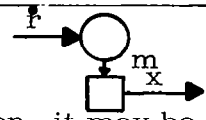
We will next argue that means cannot be selected prior to identification of the prime functions which they implement. An intuitive demonstration will suffice. Thus, given a block of cement as a candidate means, one has no criterion for determining what attribute of the block to measure in order to determine its suitability as a means unless the function it is to perform has been identified.

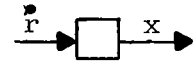
If the block is to support a mass, it will be important to test its strength under compression. If it is to be used as a counterweight, one should be concerned instead with its weight. Thus, the criteria for selecting means are not available until prime functions have been identified, and means cannot be selected before prime functions.

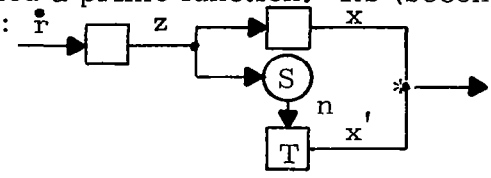
Our third argument is that second-order additive loops cannot be selected prior to first-order additive loops. This argument is readily reduced to the first argument presented above. Thus, any additive loop may be expressed as a function.[1] As a single function, it may be treated as a prime function. But in the first argument we showed that an additive loop cannot be designed prior to its prime function. Therefore, if we express an additive loop as a single prime function, then it is clear that its additive loop cannot be selected prior to selection of its means of implementation. Therefore, a second-order additive loop cannot be identified prior to the design of its first-order additive loop. We will generalize this argument to the additive set and say that the second-order additive set cannot be stabilized prior to the first-order additive set. Or, even more generally, that the n plus first additive set cannot be stabilized prior to the stabilization of the nth additive set.

Our final argument is that all of the models and tools needed to enable fabrication of the physical operational system cannot be stabilized prior to stabilization of the complete functional and means design of the operational system. The verity of this assertion is self-evident. There is no basis in the state of the art for prescribing how a part will be fabricated until the part to be fabricated has been identified in at least one of its aspects.

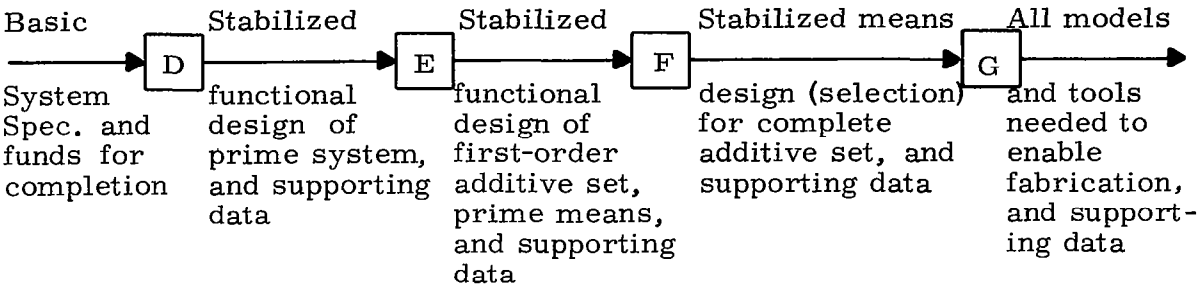
---

[1] Thus, one may add 

to attain the function 

As an isolated function, it may be called a prime function. Its (second-order) additive loop, S, T, would be arrayed: 

We now present a symbolic definition of the four functions in the index model which are the components of Phase II:



It can be seen that this portion of the index model provides for a sequence of steps that is in agreement with the arguments presented above. This part of the model implies that if there is to be a stepwise order, then the output of G must follow the previous outputs in sequential order, $D \rightarrow E \rightarrow F \rightarrow G$.

As shown in the diagram above, the output of Function D is a stabilized functional design of the prime system and supporting data. All of the output states shown are two-part states, one part being a stabilized design, the other part being supporting data. As described above, the supporting data result from trial or representative design actions undertaken for the purpose of determining feasibility. The information contained in supporting data must be discriminated from stabilized design data which are presented with the understanding that they will not be changed later. The cumulated stabilized design data are given to all participants in a development cycle and form a stable basis for further design actions; the supporting data do not.

The output of Function E is a stabilized functional design of the first-order additive set, stabilized identification of prime means, and supporting data.

The output of Function F is an accumulation of all previously stabilized design information plus stabilized identification of the means for the complete additive set, and supporting data.

The output of Function G is the output of Phase II. It includes all stabilized design information accumulated in Functions D, E, and F, as well as identification of the models and tools needed to enable fabrication of the operational system identified in the design solution. The output of this sequence of four functions is thus a collection of models including: parts lists, wiring diagrams, installation diagrams and instructions, engineering drawings, fabrication instructions, specifications for components to be purchased off-the-shelf, and so on. In fact, the collection includes all of the models necessary, whatever they may be, to ensure the successful fabrication and installation of the means that will implement the operational system.

Function D

The key input to this function is the Basic System Specification. Its output includes two key elements: (1) a stabilized functional design of the prime system, and (2) data to demonstrate that there is at least one system solution in the target Cost, Quality area within the family of solutions encompassed by the stabilized prime functional design.

The output state of Function D is a necessary one within Phase II. Much of means selection in system development cycles is carried out against implicit functional criteria, but whether the criteria are implicit or explicit, they are, as we have shown, absolutely necessary precursors to means selection. Stabilization of the functional design is therefore a necessary state in Phase II.

The order of the output state of Function D with respect to other output states in Phase II has already been discussed.

Within Function D, the designer's task is to pick a "winning" functional design with as few false starts as possible. A winning design will be one for which supporting data can be provided. In order to develop supporting data it will be necessary for the designer to review all of the remaining steps in the development cycle that will have to be undertaken if the suggested functional design is stabilized. Thus, the designer will have to show that Functions E, F, G, and H can all be carried out with success. The development of such data is a large undertaking requiring a good deal of "representative" design effort. Therefore, it is important within Function D to take steps to avoid estimating design, Costs and Quality for a large number of alternative functional designs.

Function E

The output of Function E includes: (1) a stabilized functional design of the first-order additive set, (2) a stabilized identification of the prime system means, and (3) data to demonstrate that the stabilized design decisions made in Functions D and E do not exclude the possibility of completing the development cycle to achieve a system in the target Cost, Quality neighborhood. The input to Function E is the output of Function D, a stabilized prime functional design.

Identification of prime system means is a necessary precursor to the preparation of fabrication instructions and fabrication tools. The output of Phase II includes such fabrication instructions, and therefore the output of Function E is a necessary state within Phase II.

The relative position of Function E with respect to other functions in Phase II has already been justified.

For each prime function identified in Function D, a means will be selected and stabilized in Function E. In many cases, the reliability of the means selected to implement a function will not be sufficient to satisfy its probability of output requirement, and it will be necessary to provide for an additive loop to make up the difference. When there are alternative means by which a

prime function may be implemented, selection may depend not upon comparison of the means alone but upon joint consideration of each means and the additive loop that is its adjunct. It can be seen then that Function E will, in a sense, be a complex trade-off exercise focused upon achieving the best joint selection of prime function means and their associated additive loops such that the overall selection across the board can be justified by supporting data as leading toward an acceptable system solution.

The joint consideration of alternative combinations must be undertaken in a manner which considers the additive set as a whole, for in the implementation of the additive set to be undertaken in Function F, frequently it will be desirable to employ multipurpose means which cut across additive loops. (A maintenance technician is an example of a multipurpose means that is employed in implementing additive loops.) A good solution for an additive loop taken by itself may not be good when the total additive set is considered.

Function F

Function F produces a stabilized means design for the entire operational system, including the complete additive set. The means design that is stabilized must be accompanied by data which demonstrate that the physical system identified falls in the target Cost, Quality area. The key input to Function F is the functional design of the first-order additive set; it is the means for implementing this functional design that is a principal concern of Function F. The function also encompasses design of all lower-order additive loops.

The output of Function F identifies system means and is therefore a necessary precursor to Function G. The completion of Function G requires that all system means previously be identified.

Activities in Function F include functional identification of second-, third-, and lower-order additive loops as needed to provide for overall probability of system success. They also include the selection of means for implementing all additive loops. As in the case of Function E, additive loop design must be done on the basis of consideration of the additive set as a whole.

Function G

The output of Function G is the output of Phase II. It includes: (1) the set of all fabrication models necessary to enable the manufacture, assembly and installation of a satisfactory operational system, (2) all of the special fabrication tools needed for fabrication, (3) data to show that the specific operational system described by the set of fabrication models will be one which falls in the target Cost, Quality neighborhood. Its input is the output of Function F.

The rationale for Phase II supports the requirement for the output state of Function G and demonstrates that it must precede Phase III.

Within Function G, sufficiently detailed models must be developed such that physical means may be selected, fabricated or otherwise developed. Within Function G concern will focus upon determining that the means described are indeed physically realizable and upon assuring that they may be articulated in a manner that will yield a completely integrated physical system of satisfactory Cost and Quality.

Phase III, Component Function H

Phase III is not partitioned in the index model; it is simply renamed Function H. The failure to partition Phase III reflects a study decision to focus upon development cycle activities prior to fabrication efforts. This stand was adopted because there is greater apparent need for detail in Phases I and II; the current state of the art for carrying out fabrication is relatively superior and much better known.

Function H accepts the fabrication models which are the output of Function G and accomplishes the translation of these models into real-world physical means that are then delivered as an assembled operational system. The delivered operational system and data to demonstrate that it satisfies the Basic System Specification are the outputs. Inasmuch as the termination

of Phase III and of Function H coincide with the termination of the development cycle as a whole, the outputs of the development cycle are the outputs of Phase III and of Function H.

The Complete Index Model

A symbolic representation of the index model is presented in Figure 1. None of the output states given in this model is completely described. However, an attempt has been made to provide enough information about each output state such that the other elements in the output state may be inferred and filled in by the user on the basis of reason. To give more information in the symbolic model would be to clutter it unnecessarily. Furthermore, without a much more detailed consideration of the whole development process, it is probably impossible to characterize fully all of the elements in the output states.

In the model in Figure 1 the "boxes" are named. Care should be taken that these names do not divert attention from the fact that further partitioning of the functions symbolized by the boxes and their associated input-output states will be made on the basis of the input-output definitions and not on the basis of the names given within the boxes. The names within the boxes are given principally to satisfy those readers who find such names to be convenient "handles". In the full model that will be considered next, boxes will be given numbers as names rather than word names. To carry the naming of boxes beyond the index model would call for a considerable effort in formulating unique names for all of the boxes in the full model. Furthermore, experience indicates that naming the boxes detracts from the important information, the output states.

In the sections which follow, each of the functions in the index model will be partitioned in turn into component functions which will be called activities. This partitioning will accomplish the presentation of the full development cycle model. In this partitioning, focus will be upon component functions which are related to the development of personnel products within a structure which takes account of the unique aspects of aerospace systems.

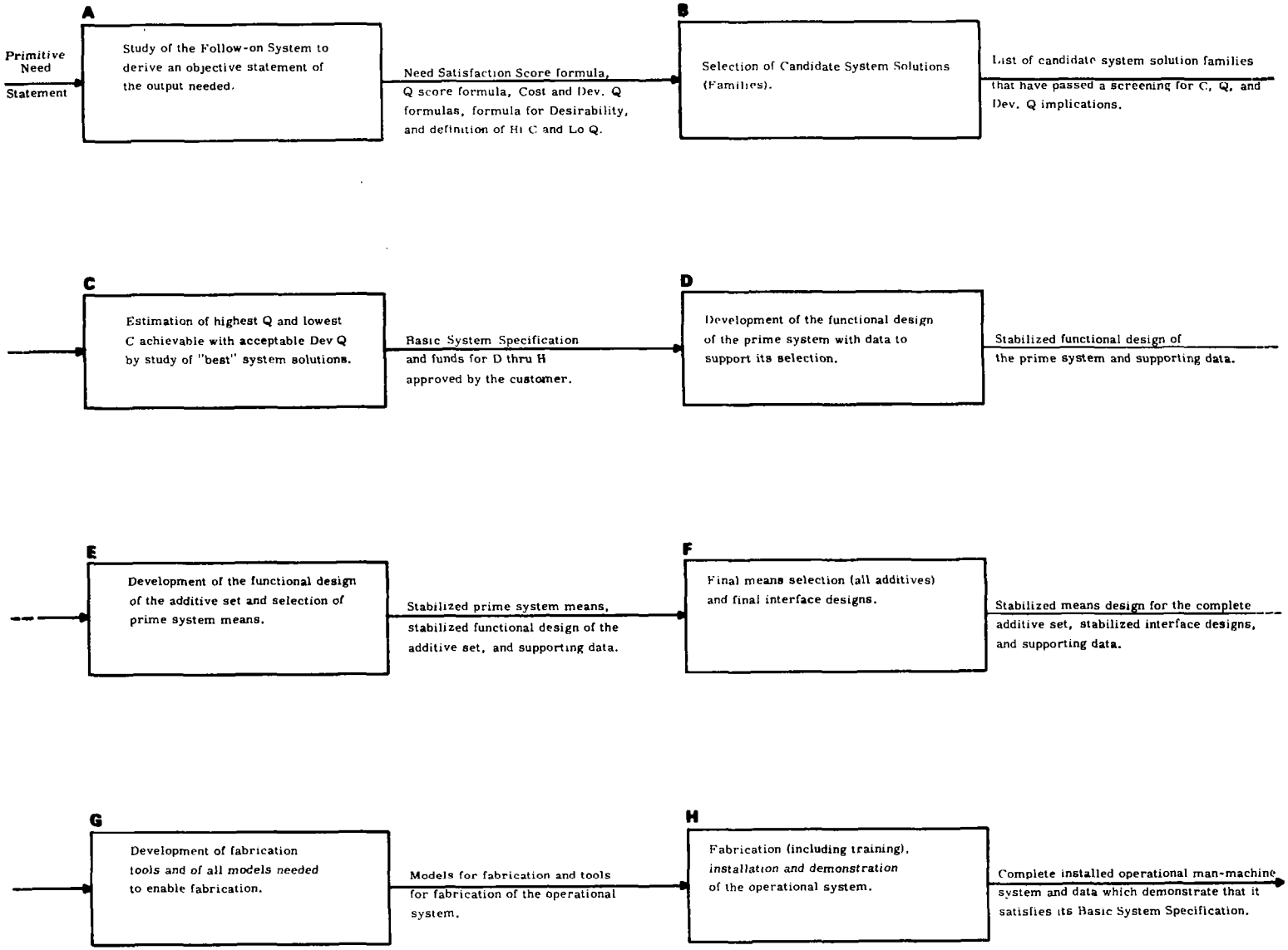


Figure 1. Schematic representation of the index model.

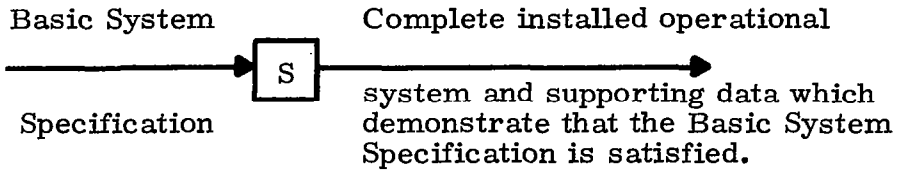
IV. THE DEVELOPMENT CYCLE MODEL

In this section we will present the full development cycle model and the rationale by which it was derived from the index model. First, there will be an overview of that part of the rationale which is common to the partitioning of several of the functions in the index model. This will be followed by eight sections, each of which will present a detailed description of the component activities of one index function. Each description is accompanied by a figure that displays the component functions of the reference function in symbolic form. The detailed description of each of the activities in the model is tedious and will be of interest only to the most determined reader. To denote its status as descriptive material included for completeness, it is presented in a compact format.

In overview of the entire model, it will be seen that the nature of the development process changes between Functions C and D, as reflected in the overall structure of the complete development cycle model. Functions A, B, and C (Phase I) stand apart because there is no external criterion for judging their output. Rather, Functions A, B, and C are concerned with the generation of criteria by which efforts in Functions D through H may be guided and evaluated. These criteria are contained in the principal output of Function C, the Basic System Specification.

Phases II and III, Overview of Partitioning

The rationale underlying the use of the Basic System Specification is employed over and over again in the partitioning of Functions D, E, F, G, and H and it will therefore be useful to consider that rationale. Fundamentally, the Basic System Specification is a <u>test</u> specification. It describes a complete and objective test by which the operational system as a whole may be evaluated, and it identifies what is meant by a "passing grade." When the Basic System Specification is conceived in this way, then we might represent Functions D through H as a single function, S, as shown in the sketch below.



In the diagram above, the function that is obtained by adding Phases II and III is bounded on the input side by a test specification and on the output side by an end product and data which demonstrate that the delivered end product has passed the "test." It is this pattern which is the characteristic pattern that will be employed over and over again in the full development cycle model—a pattern of specifying or "ordering" a piece of work by means of identifying the test which it must pass and then of bounding the output by a delivered end product plus data which show that it has passed its test.

In order that every design and fabrication activity at every level of detail in the model may be directed toward a specific and justifiable goal, the principle of specification by means of disclosing the test of the end product is employed throughout. To ensure that the tests are relevant, provision is made for all test specifications to be derived in an orderly manner from the overall system Quality score formula. No arbitrary tests which cannot be shown to be predictive of effects upon Quality have been introduced.[1] Provision for ensuring that all tests are related to Q is made by deriving all tests of system parts and subparts in an orderly progression from the overall system Quality score formula.

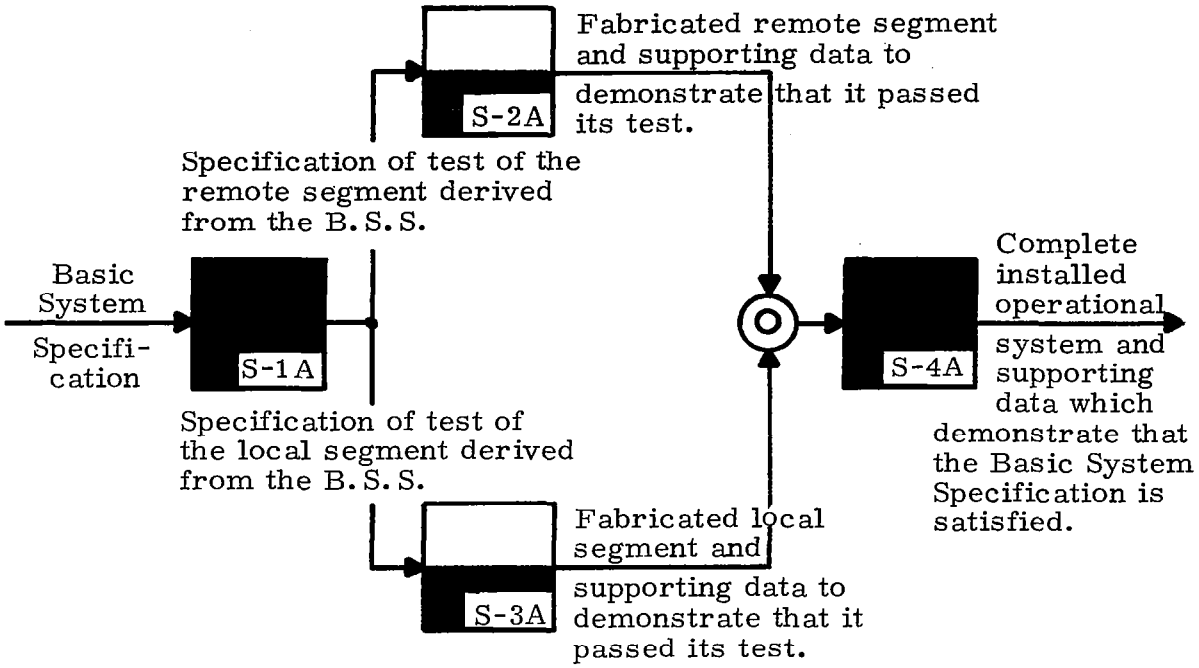
[1] The model does not identify specifically all ways in which the "A" score formula must be taken into account in the development process. All tests for quality within Functions D through H must include consideration of the "A" score formula for the system under development in the same manner as the Quality score formula is considered.

To exemplify the manner in which component Functions D, E, F, G, and H are partitioned in the full model, we will temporarily ignore the fact that Phases II and III are partitioned into these five functions and treat Phases II and III as a single function as symbolized in the schematic representation of Function S above. The partitioning of this function that we will develop will be typical of the partitioning of Functions D, E, F, G, and H in the full model.

The pattern in which Function S is partitioned is determined by the fact that aerospace system fabrication efforts are organized about physical packages. Fabrication efforts are not organized such that each subeffort corresponds to a specific system function, nor are fabrication efforts organized about specific technologies such as pneumatics, hydraulics, mechanics, and so on. They are, rather, organized to correspond to the major pieces and major subpieces of <u>things</u> that are to be delivered, assembled and installed to make up the total operational system that is wanted. Whether this method of organizing a fabrication effort is good or bad is not at issue here. It is a fact that we organize major system fabrication efforts in this manner and that this method of organization has passed the test of practice well enough to have survived.

On the basis of practice, then, we first partition Function S into four component activities as shown in the following diagram.

In the diagram, Box S-1A and Box S-4A are at the "system" level. Boxes S-2A and S-3A are at the "segment" level, the remote and local segments being the first-order breakout of an aerospace system in terms of packages. Even at this first level of partitioning, the essential nature of the array that will be developed is revealed. The input to Box S-1A is the Basic System Specification. As an "order" for system design and fabrication, it is essentially a statement of how the output of Function S will be tested when it is delivered. The output of Box S-4A is the output of Function S. It includes the delivered, installed operational system and data to demonstrate that it passes the test implied by the input to S-1A. The test may be applied again by the customer, but presumably any testing the customer might do would develop data essentially the same as the data presented as supporting data in the output of Function S-4A.



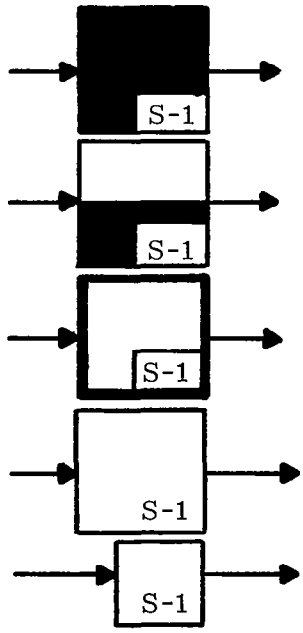
In the partitioning of Function S, this product-plus-test-data pattern is preserved at the segment level. Thus, the input to Function S-2A, for example, is a description of the test of the remote segment which the output of the activity must pass, and the output is the delivered end product (the remote segment) plus data which show that its test has been passed. The pattern is repeated again for the local segment in Function S-3A. Function S-4A is then one which assembles and tests the remote and local segments as a complete operational system.

The pattern of specifying a test and applying the test to develop supporting data is repeated at all levels in the further partitioning of Function S. At the next level of partitioning, we call out the major packages within segments which are manufactured as packages. For example, typical packages within the local segment are structures, propulsion, guidance and control, and payload. Examples in the remote segment might be propellant handling and launch platform. In the model we are presenting here, we are not concerned primarily with the specific breakout of hardware packages, however; we are interested

in typifying the hardware breakout in order to establish the level of breakout at which the personnel products package will appear in parallel. In our model we have placed the personnel products package in parallel with hardware package breakouts at the level implied above. Therefore, the next level of partitioning of Function S appears in the diagram on page 78.

In the array of activities shown, activities S-2B and S-3B develop descriptions of the tests by which the packages at the next level of breakdown will be evaluated. These tests are derived from the next higher level tests and become the "orders" for fabrication at the next lower level; thus they provide the basis for developing the supporting data which accompany each of the outputs at the next lower level.

It can be seen that the personnel products package appears at the second level of breakdown. It can also be seen that a "clam-shell" type of pattern is emerging in which activities concerned at the system level run down the middle with segment-oriented activities on either side. This clam-shell pattern will be maintained with activities concerned with the remote segment on top and activities concerned with the local segment on the bottom. The boxes are coded to imply the level of breakout as follows:



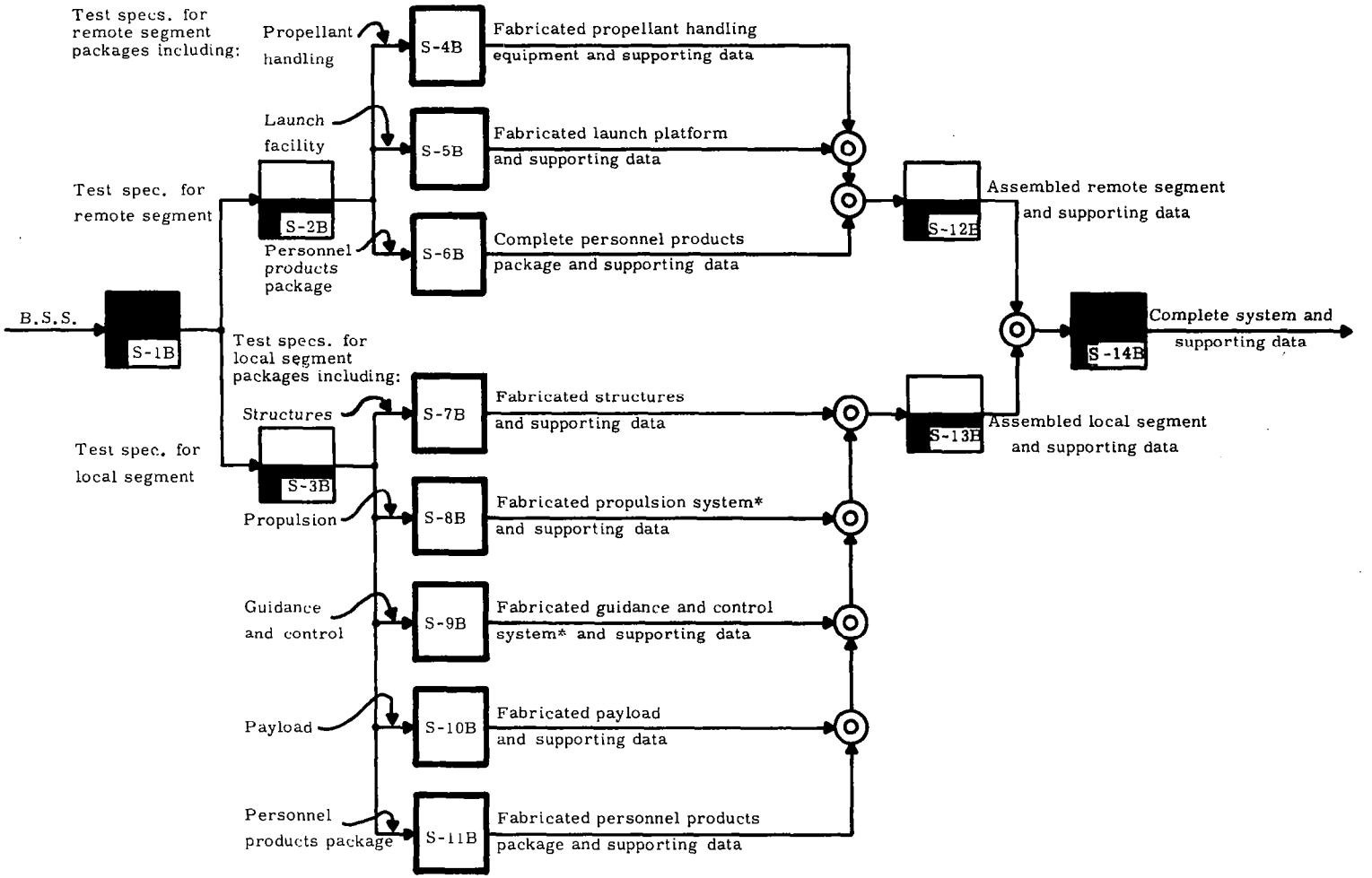
Activities concerned with dividing a system description to the segment level and with the integration of segments into a complete system.

Activities concerned with dividing a segment into packages and with the integration of packages into segments (remote or local).

Activities concerned with dividing packages into first-order component packages and with integrating these into packages.

Activities concerned with dividing first-order component packages into second-order component packages and with integrating these.

Activities concerned with second-order component packages.



* more properly "package" than system

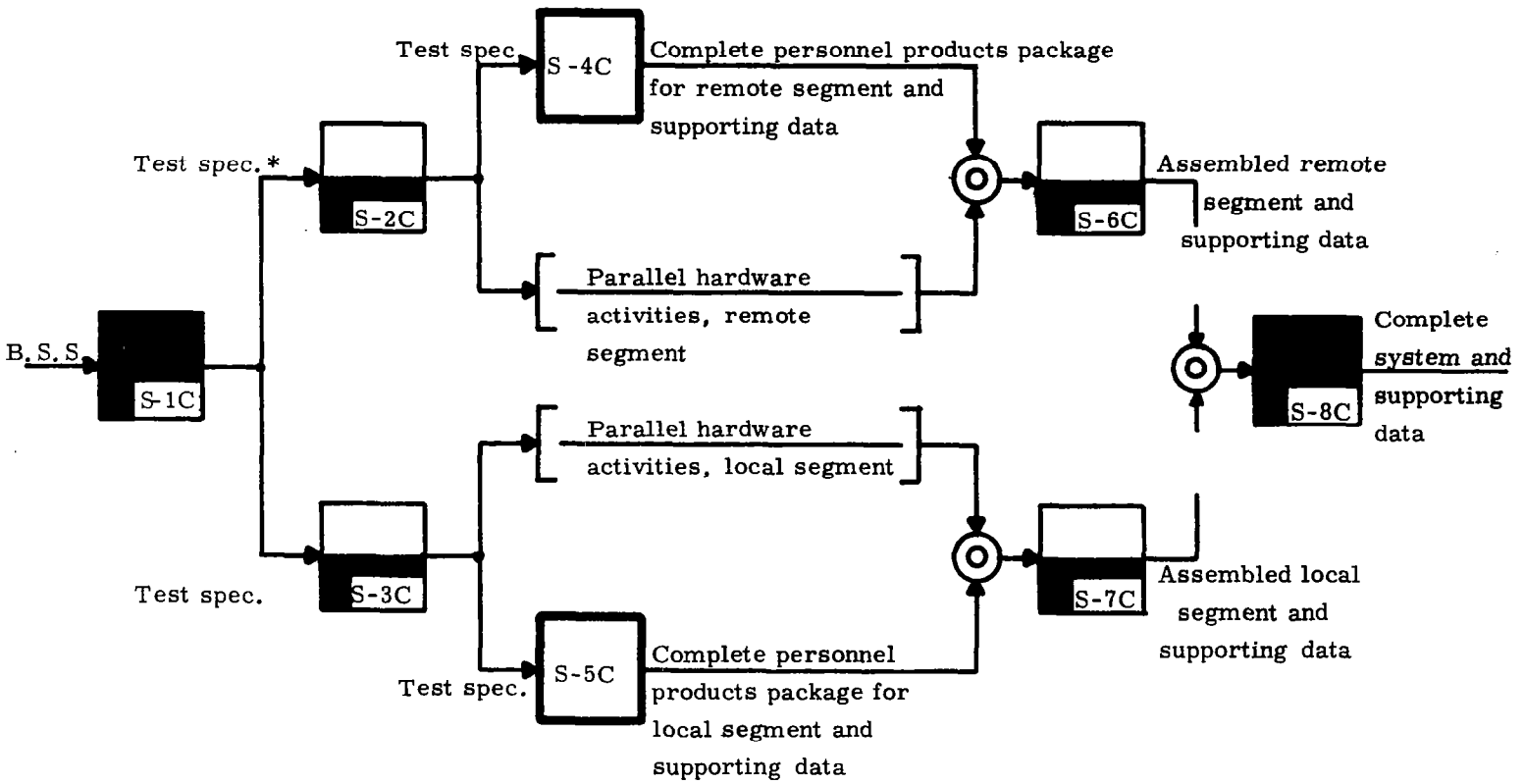
In presenting the model, the detail about hardware packages is not of use and therefore the detail will be reduced to a simple reminder, as shown in the following figure (p. 80), which should be compared with the partitioning of Function S shown above.

The form of the model for Function S shown on p. 80 is the same as the form of the model for Functions D, E, F, and G. By comparing it with the previous partitioning of S, the reader will become familiar with the shorthand conventions employed to reduce unnecessary detail in the symbolic models.

As one moves from Function D to Function H in the full model the identification of the states changes; it is the overall pattern that remains the same. In the complete breakout of activities for each of these functions, the personnel products package activity is further broken down, sometimes to one lower level of detail, sometimes to two lower levels of detail. The specific breakdown of the personnel products package for Functions D, E, F, G, and H will be presented as the partitioning of each of these functions is considered in the discussions that follow.

Phase I, Overview of Partitioning

For the most part, the activities which make up Functions A, B, and C fall in series in the full model. In these functions the model does not specifically differentiate man-related and hardware-related activities. The principal reason for this lack of differentiation is that to differentiate would force the presentation of a model at a very fine level of detail. At the level of detail of the present model, hardware-related and man-related activities are so interwoven in the prosecution of Phase I, that they cannot usefully be identified by separate activities.



*The "package" covered by the test spec. may be determined by referring to S-6C.

The Partitioning of Function A

The component activities of Function A are given in symbolic form in Figure 2. The activities shown in this figure are all focused upon study of the follow-on system. None of the component activities of Function A concerns itself with the interior design of an operational system to solve the customer's problem.

In the model the customer's problem is fully and objectively defined in three steps. The order of these steps is determined by the fact that each succeeding step requires the data produced by its predecessor.

Activity A-1

This activity is initiated by a Primitive Need Statement. Its output is a Need Satisfaction Score formula and a target Need Satisfaction Score, both of which have been approved by the customer. Occasionally, this activity may be bypassed without penalty. Thus, if there exists an adequately documented model of the customer's system, it may be possible immediately to develop a Quality score formula without going through the Need Satisfaction Scoring stage. In general, however, the development of an aerospace system is occasioned by needs in systems which have not been documented in a manner that will permit bypassing this activity.

Activity A-2

The output of this activity is a Quality score formula approved by the customer. The Quality score formula is constructed as an alternative way of measuring need satisfaction and it, therefore, cannot be developed until after the Need Satisfaction Scoring formula has been approved. Many systems have been built without an apparent formalized Quality score formula, and it may therefore be questioned whether or not such a formula is needed. A Quality score formula is a functional definition of the system that is required by the customer. As a function definition it cannot be preceded by the selection of means.[1] It must follow, then, that when systems are developed without a formal Quality score formula, there has been an implicit formula against which means have been selected. It is difficult to argue that a development cycle which employs implicit criteria will be more efficient than one which employs explicit criteria. We, therefore, show the explicit output of this activity as a necessary one in Function A.

---

[1] See the rationale for the partitioning of Phase II into component Functions D, E, F, and G.

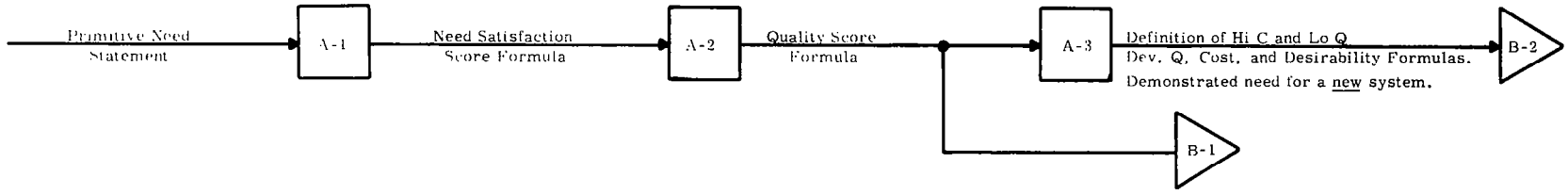


Figure 2. Diagrammatic overview of Function A (Phase I) showing the component activities and their relationships.

Activity A-3

The output of this activity is achieved primarily by working with the customer. The output includes the following:

> A statement of the formula by which costs will be determined, including provision for separate accounting of development cycle costs as opposed to operational system costs;
>
> Definition of the upper boundary of Cost for system development operation and maintenance;
>
> Definition of the boundary of Quality below which a system would be deemed to be undesirable without regard for Cost;
>
> Statement of a formula for determining system desirability on the basis of Cost and Quality;
>
> Demonstration that there is an established need for a new system (i.e., demonstration that no existing system satisfies the customer's needs).

The data developed in this activity complete the list of data required as a basis for carrying out Function B. Taken together, the outputs of activities A-2 and A-3 provide an objective basis for further development. Thus, these outputs make public the manner in which the customer will evaluate any solution that is offered to solve his problem.

The activities in Function A may require the services of human factors or biotechnological experts, especially if the problem in the customer's system is one involving personnel performance. However, there is nothing about the activities in Function A that is uniquely biotechnological. What is required to carry them out is cleverness in achieving an objective description of a problem.

The Partitioning of Function B

In presenting the rationale for the index model, Function B was included because it was necessary to identify candidate system solutions before they could be evaluated in Function C. A further burden was placed upon Function B, that of filtering out candidate solutions unworthy of consideration in Function C, so that Function C might not become "overloaded." In fact, this second part of the purpose for Function B is not a necessary one; rather it is an optional one which implies an additive function. Nevertheless, because of its importance, a typical sequence of activities to accomplish the filtering is shown in the partitioning of Function B. It should be understood that the activities which account for this filtering action are representative and cannot be justified either as necessary, nor as the best sequence of activities by which filtering might be achieved.

The component activities of Function B are shown in symbolic form in Figure 3.

Activity B-1

The output of this activity is a list of system solution families — that is, of families which are believed to include solutions that have Quality greater than zero. The level of detail of specification of the proffered system solutions cannot be prescribed. In some cases, where the state of the art offers existing systems which closely approximate a solution, the extent of detail may be great. In other cases, however, the extent of detail may be restricted to the name of a key physical process around which one might be able to build up the required operational system. Any and all candidates which may have a Quality greater than zero should be listed, for there is no other function in the development cycle which specifically calls for the introduction of new candidate system solutions.

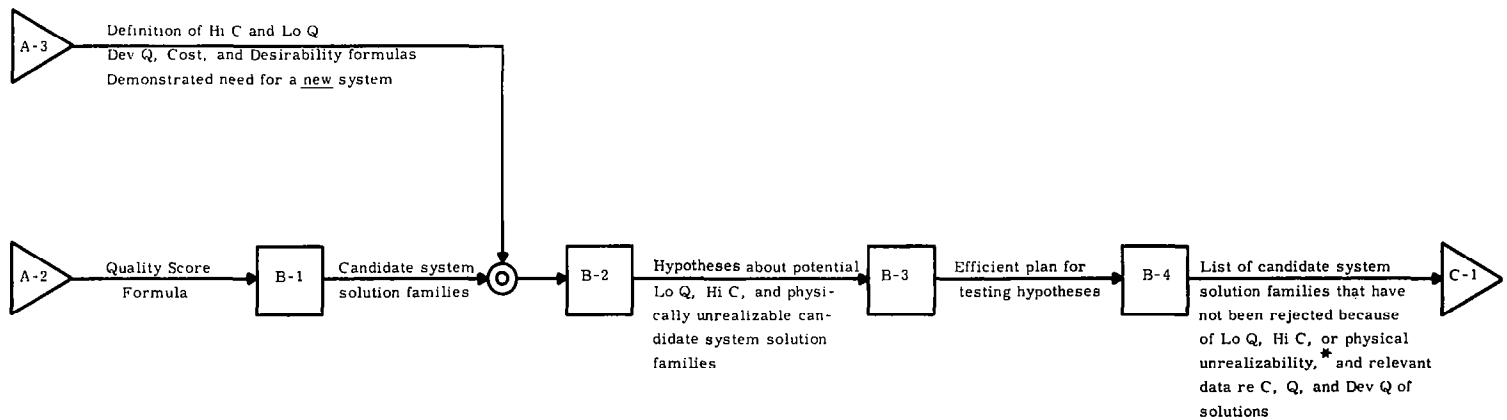
The input to this activity is the Quality score formula which is the output of activity A-2. The output of activity A-3 is not shown as an input. This emphasizes that activity B-1 is focused upon listing candidate system solutions and not upon evaluating them. Before a candidate can be evaluated it must be listed. The criteria necessary for a complete evaluation are developed in activity A-3 and are not needed as inputs to activity B-1.

Activity B-1 is the only activity in Function B that is absolutely necessary as a precursor to Function C. The activities which will be discussed below are those which typify a sequence to accomplish "filtering."

Activities B-2, B-3, and B-4

The objective of this sequence of activities is to remove from further considerations any system solutions suggested in Activity B-1 that can readily be shown not to be satisfactory — because of Cost, because of Quality, or because of poor expectation that they can be physically realized (low Dev Q). Inasmuch as the justification for this kind of filtering is the husbanding of resources for carrying out Function C, it is clear that filtering should stop when the cost of further filtering becomes too great as compared with the cost of implementing the Function C approach.

The basic plan of these three filtering activities is as follows: In activity B-2 hypotheses are formed about "suspect" candidate system solutions generated in activity B-1. Attention is restricted to easily tested hypotheses about undesirable solutions. Thus, as appropriate, families are hypothesized to contain only solutions of unusually low Quality, of unusually high Cost, or of unusually poor likelihood of development. In Function B-3, an efficient plan is developed for testing the hypotheses. Such planning will be desirable because some hypotheses will be subordinate to others. That is, by generating



* Note: The type of activity sequence exemplified by B-2, B-3, B-4 may be repeated several times in order to shorten the list of candidate solutions sent to C-1. On the other hand, this sequence may be bypassed altogether if the output of B-1 does not need screening.

Figure 3. Diagrammatic overview of Function B (Phase I) showing the component activities and their relationships.

data sufficient for rejection/acceptance of some hypotheses, tests of other subordinate hypotheses will be rendered unnecessary.

Activity B-4 is concerned with the collection of data according to the plan for testing hypotheses. On the basis of data collected, poor candidate system-solution families are rejected so that the output of activity B-4 is a subset of the output of activity B-1. Specifically, it will be a subset of the candidate families which cannot be easily identified as unworthy of consideration in Function C. It is noted in the symbolic model that the sequence B-2, B-3, and B-4 may be repeated as many times as can be justified until it is no longer efficient to employ the strategy of removing candidate system-solution families from consideration.

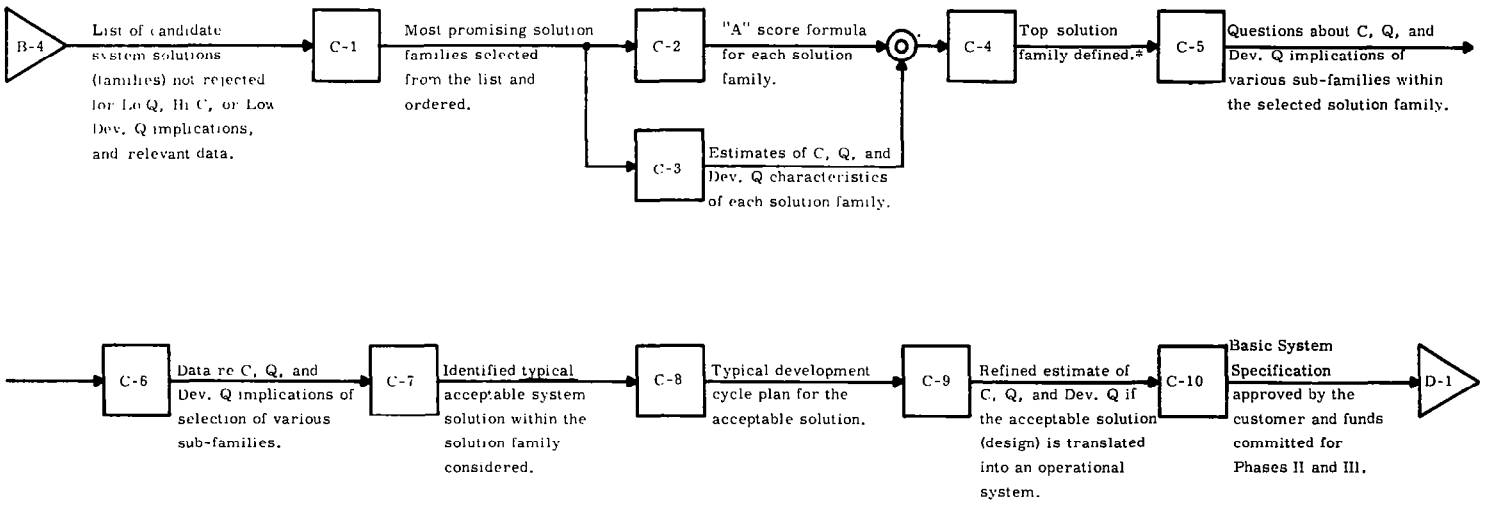
The Partitioning of Function C

The component activities of Function C are given in symbolic form in Figure 4. All of these activities are focused upon achieving a customer approved Basic System Specification and authorization of funding for Phases II and III. The basic strategy in Function C is the opposite of that of the "filtering" activities in Function B where concern is with identifying "bad" solution families. In Function C the sequence of activities is designed to find and consider the "best" candidate system-solution families. To accomplish this, the candidate system solutions are ordered, taking into account not only the predicted Cost and Quality of each system-solution family, but also the "A" score of each family. Therefore, within Function C there is concern with the development of "A" score formulas.

After the top system-solution family has been identified, Function C activities are focused upon determining the Cost, Quality, and development quality implications of the alternative subfamilies within the family, so that a representative "best" system solution may be identified. A typical development cycle plan for the "best" system solution is then prepared to provide a basis for a refined estimate of the Cost and Quality and feasibility of the representative solution — or, more properly, of the best solutions within the representative solution family. The data thus developed provide the essential information that is necessary for a presentation to the customer that will support a recommendation for carrying out Phases II and III. The data are also sufficient to prepare a Basic System Specification for his approval.

Activity C-1

The input to this activity is the list of candidate solution families which survives Function B. Its output is an ordering of these which places on top the ones that are estimated to be the most desirable on all counts. Inasmuch as the system solutions identified at this time will be families of solutions rather than individual solutions, it will be necessary in the output of this activity to fully characterize each family.



* Note: If it is desired to explore the Cost, Quality space adequately in Phase I, it will be necessary to identify several "top" solution families and to carry out parallel studies by replicating Functions C-5 through C-9 for each family. Such an approach would require comparison of the data for each family in C-10 as a basis for preparing the Basic System Specification.

Figure 4. Diagrammatic overview of Function C (Phase I) showing the component activities and their relationships.

Activity C-2

The output of this activity is an "A" score formula for each high-rank solution family. To develop an "A" score formula for a solution family requires a rather significant effort, and it is therefore desirable that "A" score formulas be prepared only for those families that are worthy of detailed consideration.

Activity C-3

This activity is carried out in parallel with activity C-2. It is focused upon obtaining Cost, Quality, and feasibility estimates for each of the system-solution families in the list of "best" ones. As in the case of the "A" score formula, a significant effort is required to obtain these estimates and such estimating should be restricted to the serious contenders in the list of candidate system solutions. It is for this reason that activity C-3 follows a preliminary ordering in activity C-1.

Activity C-4

This activity requires that an "A" score formula and that Cost, Quality, and Dev Q estimates for each solution family be provided. The output of the activity is an identified top solution family. Thus, within this activity solutions are compared not only in terms of Desirability based on Cost and Quality but also on the basis of physical realizability and the predicted "A" scores for each of the best solution families considered.

Alternative top system solutions may be selected in this activity. Thus, the output of activity C-4 may be two, or even more solutions, to be subjected to further examination in Function C. When more than one system solution is given in the output state of C-4, an activity sequence such as the sequence C-5 through C-10 would be carried out for each. Parallel sequences of activities of this type would generate good information about the population of solutions in the Cost, Quality space. Examination of the resulting picture of the Cost, Quality space would be made in activity C-10 prior to the preparation of the Basic System Specification. In the diagrammatic model, however, it is assumed that only one system-solution family is identified in the output state of activity C-4.

Activities C-5, C-6, and C-7

This group of three activities is best discussed in the light of the overall purpose of the group. Taken together, it is the objective of this group to refine the candidate system solution which is identified in the input to activity C-5 by studying the various subfamilies and sub-subfamilies within the family

to determine which of these contain solutions in the target Cost, Quality area. This study will accumulate the data necessary to enable identification of a representative "best" system solution in activity C-7.

Within the group, activity C-5 is focused upon raising questions about the various subfamilies and sub-subfamilies within the system-solution family of interest. Activity C-6 is designed to develop answers to the questions raised. The end result of the studies will be data to be considered in activity C-7 to achieve an identification of a typical "best" system solution. The system solution identified will indicate the subfamily and the sub-subfamily, etc. (within the system-solution family identified in the output of activity C-4) which can be shown to yield a highly desirable operational system.

Activity C-8

This activity provides for the generation of a typical development cycle plan for the system solution identified in the output of activity C-7. The purpose of the development plan is to provide a basis for estimating the Cost of developing the typical "best" system.

Activity C-9

In this activity, the development cycle plan and the data germane to the typical "best" system solution are considered for the purpose of obtaining a good estimate of the Cost and Quality of the system solution and an estimate of the Dev Q of its development cycle.

Activity C-10

The data generated in activity C-9 taken together with selected data generated in previous activities will enable the development of a Basic System Specification which may be submitted to the customer for approval along with a recommendation for funds to be committed for the development of a system. The output of activity C-10 is an approved Basic System Specification and a commitment of funds for Phases II and III of the development cycle.

Phases II and III

As noted earlier, the remaining functions in the index model are carried out under the Basic System Specification generated in Phase II. It will be seen in what follows that the typical plan described earlier is repeated in Functions D, E, F, G, and H. In the discussions which follow we will focus

attention upon those activities in the model that are most closely related to the development personnel products. The other activities in the model will be considered only in overview.

In approaching Phase II, it should be remembered that the output state of Phase I need not identify the specific system solution to be employed in Phases II and III even though it will be identified in practice. Therefore, the first function in Phase II, Function D, must be concerned with selecting the specific system solution of choice within the criteria set forth in the Basic System Specification. When the choice is given in the Basic System Specification, this need not be done.

The Partitioning of Function D

The components of Function D are given in symbolic form in Figure 5. The activities in this function are focused upon achieving a stabilized functional design of the prime system which is integrated at the system level and which provides a firm basis for further design activities in Phase II. The apparent simplicity of the array of functions shown in Figure 5 can be misleading. Taken together, the component activities of Function D demand extensive resources for their prosecution. Thus, the principal output of Function D, a stabilized functional design of the prime system, can be achieved only when there are data which support confidence in the decision to stabilize the functional design. To obtain the supporting data, it is necessary that the component activities of Function D fully explore the kinds of activities that must be undertaken in Functions E, F, G, and H. It is the complexity of this exploration which is hidden by the simplicity of the symbolic model.

Much of the hidden complexity may sometimes be in activity D-1. If the Basic System Specification which initiates Function D does not constrain design to a specific type of system solution, then activity D-1 may include a recapitulation of the type of function sequence which makes up Function C. In practice, however, such a recapitulation would be an exception, for in order to ensure high probability of success for the development cycle, the preferred system-solution family will ordinarily be given in the Basic System Specification.

The output of activity D-1 could, in a sense, be characterized as two complementary Basic System Specifications, one for the local segment and one for the remote segment. Thus, the output of D-1 provides the basis for an ultimate "test" for each segment.

Activity D-10 accumulates and integrates the design work at the segment level and produces as its key output a stabilized functional design of a complete prime system. This stabilized functional design provides a firm basis for further design work. It is stabilized in activity D-10 only when it can be shown that subsequent design and fabrication efforts can be carried out without driving

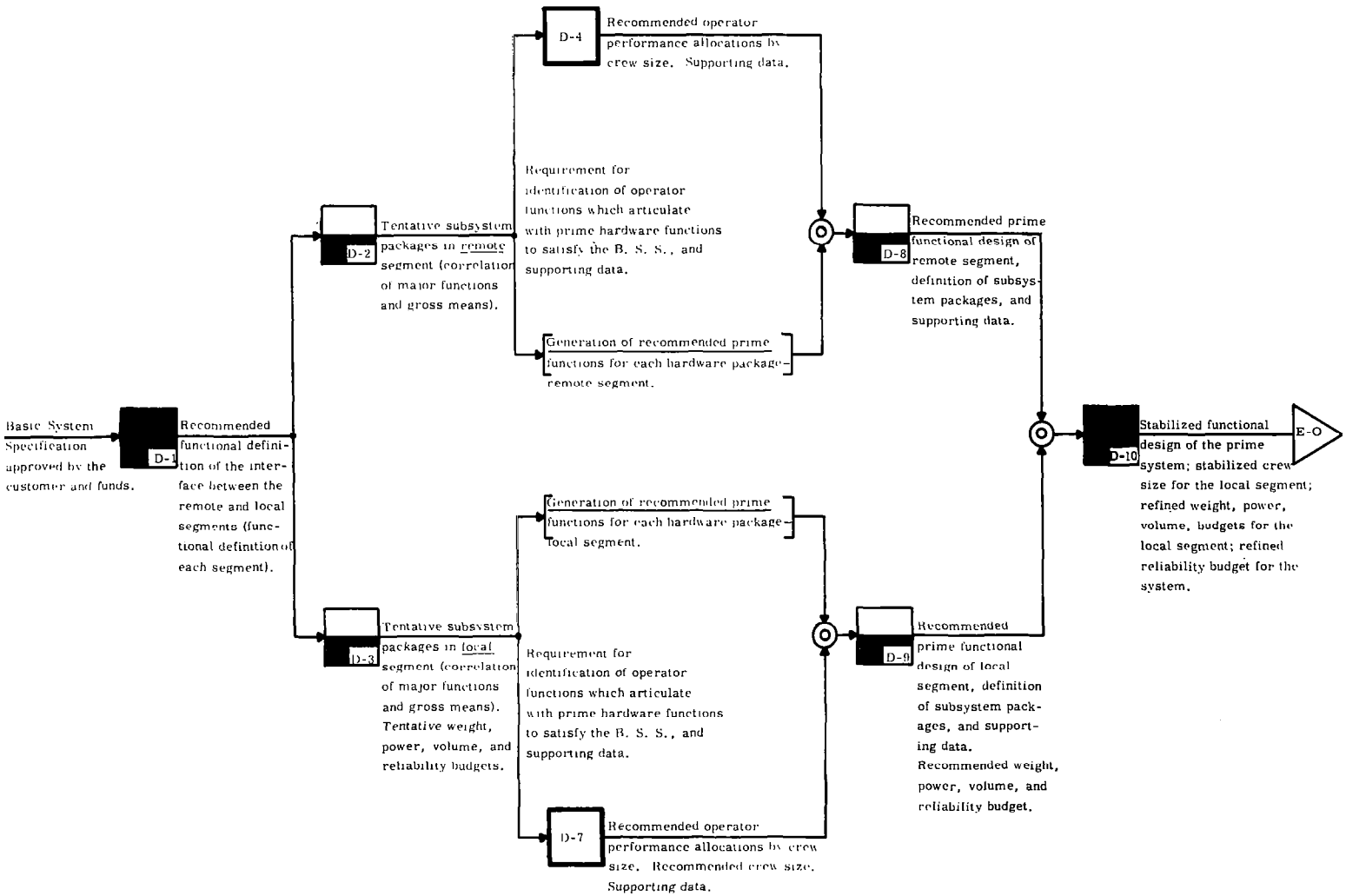


Figure 5. Diagrammatic review of Function D (Phase II) showing the component activities and their relationships.

the eventual system solution out of the expected Cost, Quality area. Inasmuch as we are concerned with an aerospace system, it is necessary also that the output of D-10 stabilize the crew size for the local segment, in order to enable propulsion system design to proceed apace, and in order to ensure that this determination is made soon enough in the development cycle to enable personnel products to be delivered in concert with hardware end products. In an aerospace system, Quality is intimately tied to concern with weight, power, and volume restrictions on the local segment. It is therefore also necessary in phase D to be concerned with these attributes of the local segment as well as with a reliability budget for the entire system.

Activities D-2 and D-3

These activities are at the segment level. They produce specifications for subsystem packages which are analogous at the package level to the Basic System Specification for the entire system. D-2 and D-3 differ in that D-3 must be concerned with weight, power, and volume budgets. In each case, one of the package specifications is a specification for a personnel products package. Basically, this specification sets up activities D-4 and D-7 so that it can be determined whether or not there will be personnel products as an output of the development cycle.

Activities D-4 and D-7

Both of these activities are concerned with identifying the prime functions to be implemented by man. Inasmuch as alternative prime functional designs may be under exploration, it will be necessary to consider many alternative manning solutions and to provide estimates of the "costs" associated with each alternative of interest. Both activities must be carried out in concert with the parallel hardware package activities and both must recommend operator performance allocations compatible with the hardware recommendations such that when all recommendations are taken together in activities D-8 and D-9 they will be found to satisfy requirements for the remote segment and local segment.

In the case of activity D-7, ordinarily it will be useful to present recommendations parametrically showing operator performance allocations and costs over a range of crew size. The same kind of presentation will be desirable in the output of activity D-4, but the burden of proof rests somewhat more heavily in the case of D-7. Thus, one target of Function D is the stabilization of the crew size for the local segment, and to achieve this stabilization, complete supporting data are necessary in the output of activity D-7. (Stabilization for the remote segment is desirable but not mandatory.)

Activities D-8 and D-9

These activities integrate the recommendations and data provided by the personnel products activities and hardware package activities and produce

recommended prime functional designs for each segment in order to satisfy the requirements placed upon the segments in the output of activity D-1.

The Partitioning of Function E

The key objective of Function E is to achieve simultaneously a stabilized means design for the prime system and a stabilized functional design for the additive set. Examination of everything that must be done to complete the development cycle so that all end products are delivered "simultaneously," reveals that Function E must also encompass design of the personnel support systems. In order that personnel support system design may "catch up," both the complete functional design and the prime means design are provisioned within Function E. To maintain a proper pacing of the personnel products activities including training, certain maintenance technician performances are allocated in Function E, even though the allocation of hardware means in the additive set is not accomplished until Function F.

In general, the configuration of activities at the system and segment levels is similar to the configuration of activities in Function D.

Activities E-3 and E-4

These activities respond to requirements from the segment level and in turn develop requirements for activities within the framework of the personnel products package. Both E-3 and E-4 develop functional descriptions of the operator performances assigned in Function D down to the level of detail necessary to enable their follow-on activities (E-5 through E-12) to carry out their assigned tasks. In the case of activity E-4, there is also concern with weight, power, and volume budgets. Basically, the requirements placed upon activities E-3 and E-4 are for the design of personnel products to achieve system probability of success. They are answered by four different types of activities. Thus, in E-5 through E-12 probability of success goals are achieved by:

1. Implementing functions in additive loops for prime hardware by means of maintenance technician performance.
2. Providing for additive loops on operator performance.
3. Providing for the articulation of man-machine and man-man means in such a manner that there is no loss of reliability due to failures at interfaces.
4. Providing for personnel support systems so that human performance will not be degraded because of unfavorable environmental conditions.

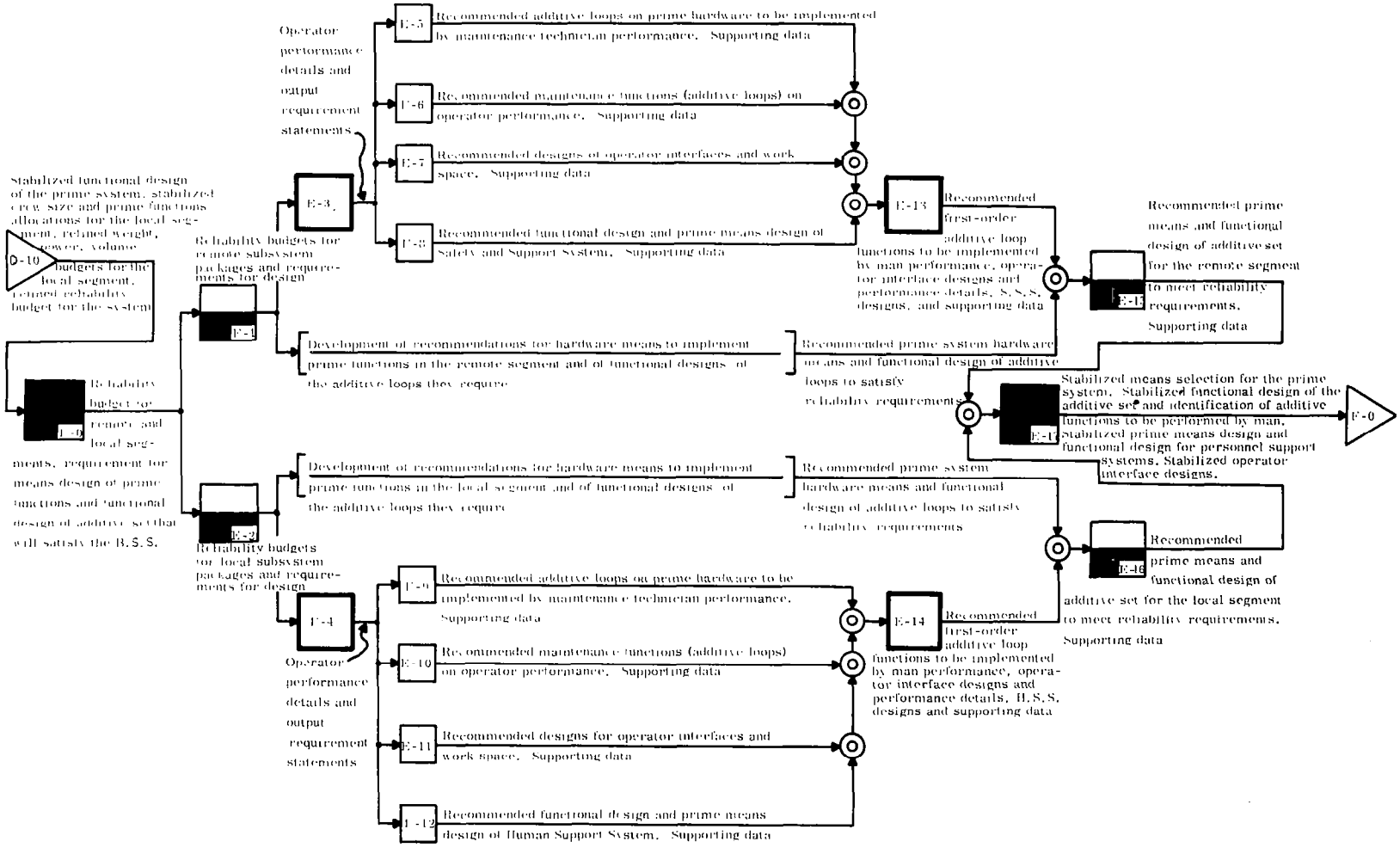


Figure 6. Diagrammatic overview of Function E (Phase II) showing component activities and their relationships.

Activities E-3 and E-4 each produce requirement statements for separate activities focused on the four methods listed above.

Activities E-5 and E-9

These activities are carried out in concert with parallel hardware activities concerned with identifying the additive loops needed to meet target reliability goals. The basic objective is to identify all of the functions within the additive loops on prime hardware which are best implemented by means of human performance (maintenance technician performance) as opposed to implementation by means of hardware. Activity E-5 is concerned with identifying the set of all additive loop functions to be implemented by maintenance technician performance in the remote segment; activity E-9 has a similar concern for the local segment. An objective of each of these activities is to present data to support the recommendations with respect to the allocation of additive performances to man.

Activities E-6 and E-10

Just as it is important to augment the reliability of prime hardware by means of additive loops, so it is often necessary to augment the implementation of prime functions by means of additive loops when the functions are carried out by operator performance. Activities E-6 and E-10 are concerned with identifying the need for such additive loops on operator performance, for specifying them functionally, and for determining which of the component functions in the additive loops on operator performance should be carried out by man—that is, by maintenance technician performance. Activities E-6 and E-10 relate to overall system probability of success in the same manner as activities E-5 and E-9.

Activities E-7 and E-11

To achieve overall system reliability, attention must be paid to the reliability with which means packages are articulated as well as to the reliability with which each means package performs its assigned functions. In these two activities, concern is with the articulation of operator functions and prime hardware functions, and the articulation of operator functions performed by one man with operator performances carried out by another. It is necessary that operator interface design recommendations be achieved in Function E because prime hardware is selected in Function E and selection must take into account the articulation of prime hardware and operator performance .

Activities E-8 and E-12

These activities are focused upon design of the personnel support systems; they are thus concerned with providing conditions necessary to sustain reliable human performance. In order to bring the design of the personnel support systems into proper phasing, these activities carry out both functional design and prime means design. The design of the Human Support System for the local segment is, of course, constrained by weight, power, and volume considerations; such restrictions are not ordinarily placed upon the Safety and Support System.

Activities E-13 and E-14

These activities assemble the outputs of activities E-5 through E-12 in a manner that will satisfy the requirements placed upon the personnel products packages by activities E-1 and E-2. In activity E-14, it must be shown that the recommendations made fall within the stabilized crew size, and within weight, power, and volume limitations. Both activities are concerned basically with demonstrating that reliability budgets for the personnel products package are satisfied.

The Partitioning of Function F

Conceptually, Function F is best seen as being concerned with the means design of the additive set. The requirements for such design at the segment level are contained in the output of activity F-0, and the response to these requirements is integrated by activity F-15 to produce the output state of the function. Activities F-1, F-2, F-13, and F-14 are the activities at the segment level; their role is similar to the role of the analogous activities in earlier functions. The important functions within the personnel products package are concerned with the identification of maintenance technician performance in all levels in the additive set, and with the design of the man-machine interfaces which result when maintenance performance is assigned to man.

Activities F-3 and F-4

These activities are at the personnel products package level. They provide detailed information with respect to the first-order maintenance technician performance assigned in the stabilized output of Function E. These activities also set forth the criteria by which the outputs of activities F-5 through F-10 will be evaluated. In the case of activity F-4, limitations are placed upon F-8, F-9, and F-10 with respect to crew size, and weight, power, and volume.

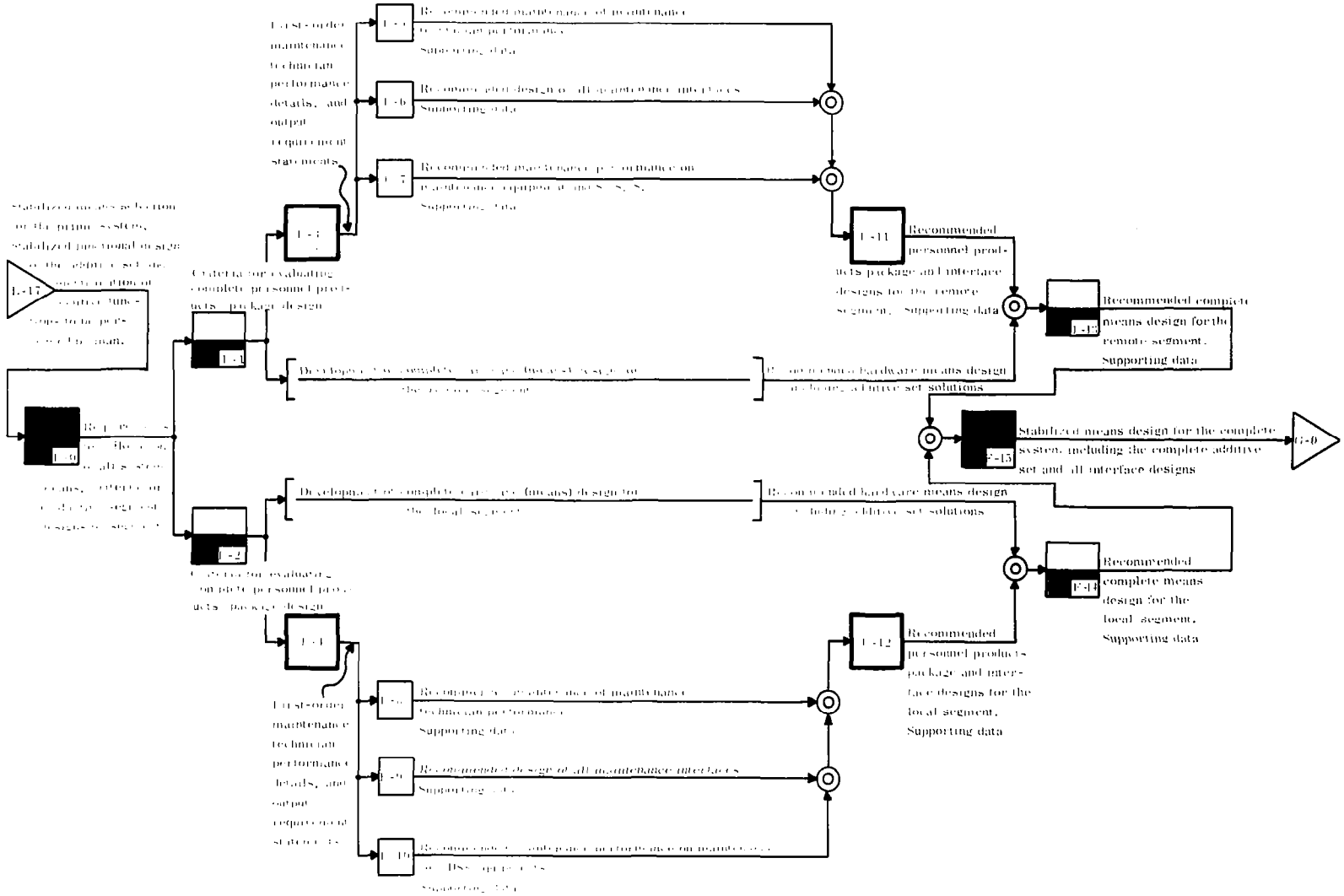


Figure 7. Diagram of overview of Function F-D used to show component activities and their relationships.

Activities F-5 and F-8

These activities provide for the identification of ways to maintain maintenance technician performance and for means to implement the maintenance functions on operator performance set forth in the output of Functions E-6 and E-10. Thus, activities F-5 and F-8 complete the provisions for the maintenance of human performance in the operational system; they complete the designs of all orders of additive loops which act on human performance.

Activities F-6 and F-9

The introduction of maintenance technician performance in order to implement functions in the additive set creates interfaces between the men who carry out the performance and the equipment upon which they act. It is necessary that the physical interfaces created by the implementation of the functional designs of the additive set be configured in such a manner that reliability of performance is not degraded as a result of interface problems. The objective of activities F-6 and F-9 is to see to it that there is no loss of overall system reliability which can be attributed to design problems at the interfaces between maintenance technician performance and hardware.

Activities F-7 and F-10

The selection of hardware for implementing the additive set often creates requirements for maintenance of the hardware itself. Thus, second-order additive loops are needed and must be implemented. Many times the implementation of second-order additive loops is carried out by maintenance technician performance. Activities F-7 and F-10 are concerned with determining which second- and third-order additive loop functions should be assigned for implementation by man. These activities are also concerned with the identification of the maintenance technician performances on personnel support equipment necessary to achieve target reliability for these support systems.

Activities F-11 and F-12

These activities serve to integrate the outputs of activities F-5 through F-10 for the remote segment on the one hand and the local segment on the other. It must be demonstrated in these activities that the total assignment of functions to personnel is commensurate with the numbers of personnel called out for each of the segments. In the case of the local segment, no adjustment of crew size can be anticipated and it is therefore necessary that the demonstration show that the crew size stabilized in the output of Function D is sufficient. It must also be demonstrated that full and proper use is

made of the entire crew. This activity must also demonstrate that weight, power, and volume allocations for the personnel products will not be exceeded. With the completion of these activities, all performance to be assigned to personnel in the system has been identified and assigned.

The Partitioning of Function G

The partitioning of Function G results in the most complex array of activities found in the model. The objective of Function G is to provide all of the fabrication plans (models) and special fabrication tools necessary to prosecute Phase III. The fabrication models and tools for those personnel products which are "things" must be prepared just as for hardware. But the personnel products package also includes crew members. For crew members, training is analogous to fabrication and, therefore, the preparation of training plans and training materials is analogous to the preparation of fabrication models and fabrication tools. Function G, therefore, includes the preparation of everything necessary to undertake training in its follow-on function, Function H.

The personnel products package includes one category of development cycle end product which violates the rule that all end products are fabricated in Function H. The fabrication of job aids is undertaken in Function G. This is done in order that the job aids may be available in Function H where they are needed as materials to be used in the training activity. An alternative would be to fabricate prototype job aids in Function G and to delay final fabrication until Function H.

Activities G-3 and G-4

The breakout of activities concerned with the personnel products package in Function G employs an intermediate level of activity not seen in previous functions (activities G-5, G-6, G-17, and G-18). The newly introduced level of activity is one which is concerned with the crew package; it excludes consideration of the personnel support systems. Therefore, the personnel products package activities G-3 and G-4 break out their requirement statements into two parts: one directed toward the crew package activities, the other directed toward the personnel support system activities.

Activities G-5 and G-6

These activities are concerned with the stabilization of the assignment of functions to the crew members by member (position make-up), and with the manner in which each of the crew members is to be given the performance capabilities required of him. Therefore, the output of these activities

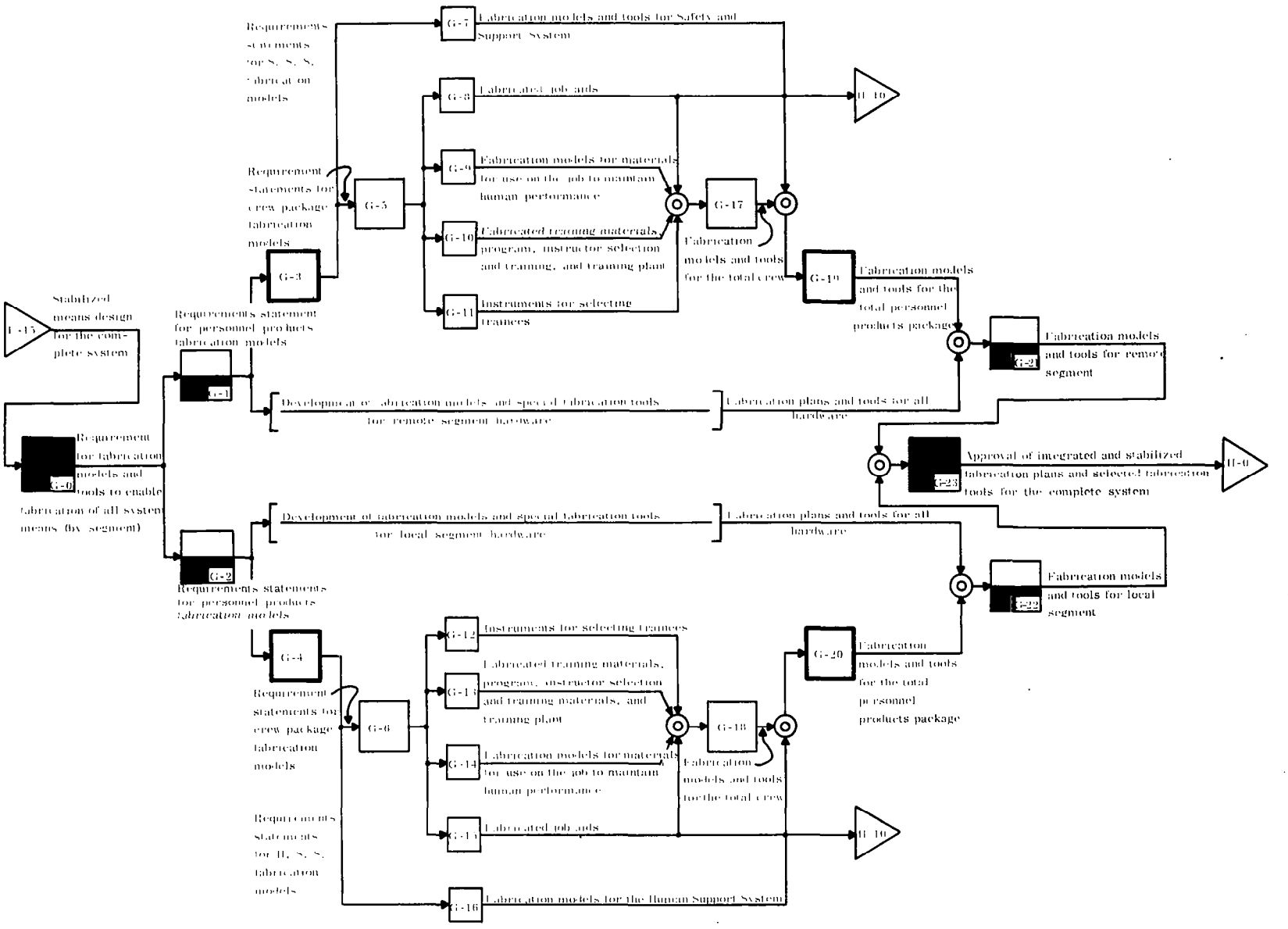


Figure 8. Diagrammatic overview of Function G (Phase II) showing component activities and their relationships.

identifies (for each crew member) the performance capabilities that will be supported by means of job aids, those that will be obtained by means of training, and those that will be obtained by means of selection. The output also includes requirements for the development of materials to be used on the job to maintain human performance.

Activities G-7 and G-16

This pair of activities is concerned with the Safety and Support System on the one hand and the Human Support System on the other. The activities provide for the development of fabrication models and fabrication tools and thus prepare for fabrication in Function H.

Activities G-8 and G-15

These activities are the ones which are "out of place" in Function G. They produce job aids which are also end products of the complete development cycle. The requirements for the job aids derive from activities G-5 and G-6. The requirements are stated in terms of the job performance which must be supported by job aids for each crew member. Weight, power, and volume restrictions are also placed upon the design and fabrication of these job aids.

Activities G-9 and G-14

These activities produce the fabrication models and the fabrication tools required to produce the materials needed on the job to maintain human performance.

Activities G-10 and G-13

These activities produce the training materials, the training program, instructor selection materials, instructor training materials, and the training plant required to carry out training in Function H. The input to these activities identifies the job performance of each crew member that is to be obtained by means of training.

Activities G-11 and G-12

These activities supplement the activities concerned with training materials and job aids. The input to these activities identifies the job capabilities which must be in the basic repertoire of selected trainees. The

output is not selected trainees; rather, it is the instruments necessary for selecting trainees, given candidates. These activities also provide for selection on the basis of background capabilities which suit selected men for training, and on the basis of health and anthropometric criteria in accordance with assumptions made in the design of the personnel support systems.

Activities G-17, G-18 and G-19, G-20

These are crew package activities which integrate all of the crew package materials including job aids, materials for use on the job to maintain human performance, training materials program, instructor selection and training materials, training plant information, and trainee selection instruments to ensure that the personnel products package made up of these items is internally compatible. The crew package is then integrated with the information generated in activities G-7 and G-16 with respect to personnel support systems to make up the recommended fabrication plans and tools for the total personnel package for the remote segment and the local segment.

The Partitioning of Function H

This function encompasses the fabrication (training) of the crew for the remote segment and of the crew for the local segment. It provides for the parallel fabrication of hardware so that the various "packages" which make up each of the segments may be integrated and tested in an orderly manner, and so that the total system may then be assembled, first as segments, and then as a total integrated system which may be demonstrated (tested) to develop data that show that it satisfies the Basic System Specification.

Activities H-3 and H-4

These activities set forth in final form the test criteria by which the personnel support systems and the remote and local crew packages will be evaluated after fabrication. They respond to similar test criteria set forth for each segment by activities H-1 and H-2. All of these test criteria employed derive from the Basic System Specification; criteria which are arbitrary and not related to the Basic System Specification cannot be justified.

Activities H-5 and H-6

These activities employ the selection instruments developed in Function G to provide selected trainees for activities H-8 and H-10. In addition to the men selected for training, the outputs of these activities must include data

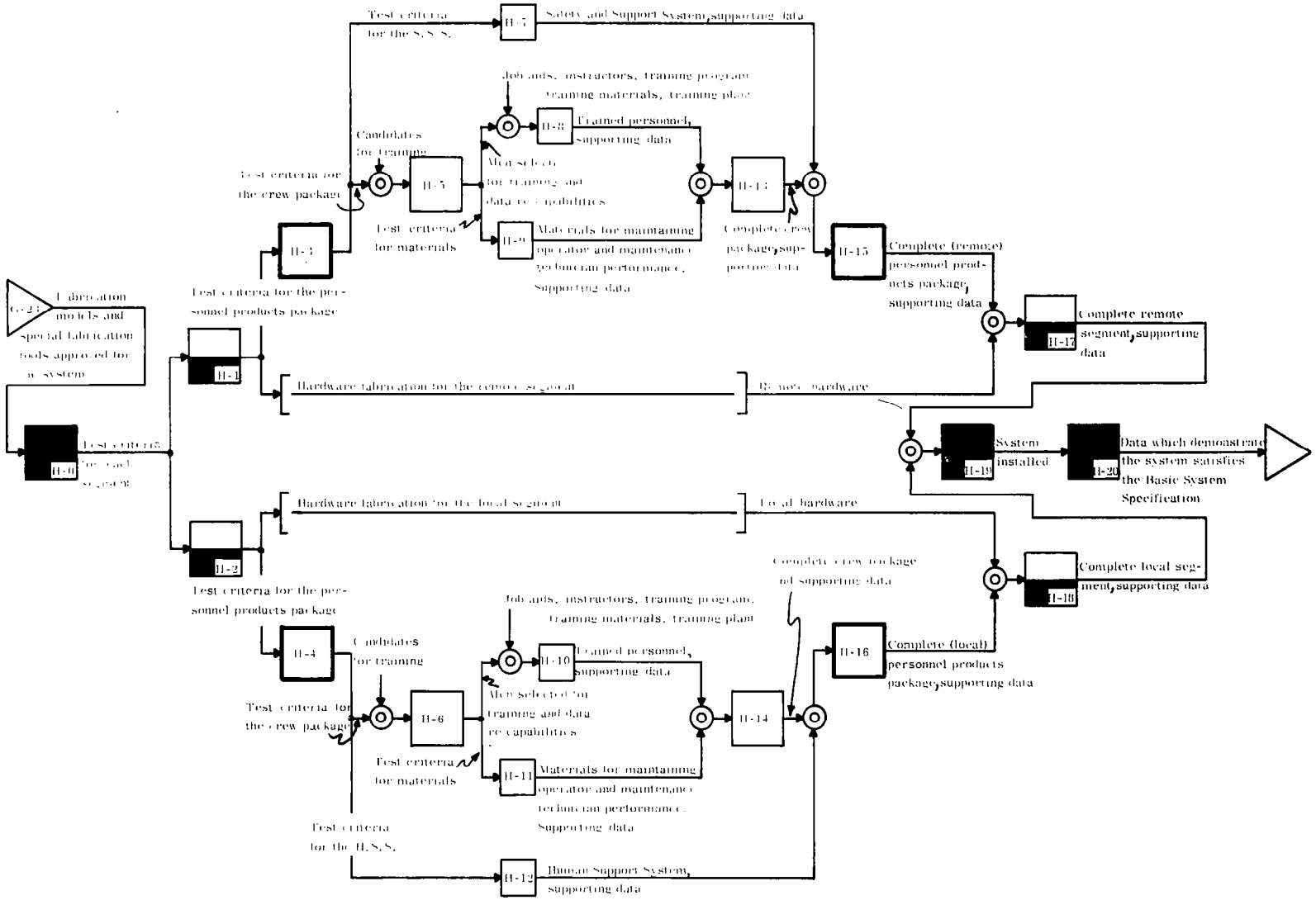


Figure 9. Diagrammatic overview of Function II (Phase III) showing component activities and their relationships.

which identify any differences between the actual capabilities of the men selected and the specifications against which they were selected.

Activities H-8 and H-10

These activities employ the fabrication tools and materials generated in Function G to train the remote and local crews. The output of each activity is a trained crew and data which demonstrate that the crew is capable of the operator and maintenance technician performances identified in the specification for the crew. It can be seen that these data are to be generated without using the personnel support systems to provide environmental conditions.

Activities H-9 and H-11

These activities produce the materials needed to maintain operator and maintenance technician performances in the operational situation. These materials are end products of the development cycle; they are not employed for the purpose of training personnel.

Activities H-7 and H-12

These activities respond to test criteria to produce the Safety and Support System and the Human Support System. These personnel support systems are elements of the operational system and are delivered end products of the development cycle.

Activities H-13 and H-14

In these activities the crew package is assembled, including trained personnel, job aids, and materials for maintaining operator and maintenance technician performance on the job. In the output of these activities it is demonstrated that these integrated packages satisfy the specification for the crew package.

Activities H-15 and H-16

In these activities the personnel products package is finally integrated and demonstrated. Demonstration includes the development of data to show that when the crew package is employed in conjunction with the personnel support systems the specification for the personnel products package is satisfied.

Activities H-17 through H-20

These activities account for the assembly, installation, and demonstration of the complete operational system. When data are generated to demonstrate that the system satisfies the Basic System Specification, the development cycle is completed.

V. USE OF THE MODEL

The model that has just been presented is one which describes the process of system development as though it were error free. There may therefore be an inclination to ask, "Of what use is a model which does not take into account the real circumstances under which aerospace systems are developed?" In part, the answer is that the model identifies a typical prime development cycle; it identifies activities which are likely to be critical to the success of any aerospace system development cycle, and which therefore must be taken into account in planning or predicting such a development cycle. The prime functions which make up a prime development cycle are as much at the heart of the matter of development cycle design as are the prime functions which make up a prime operational aerospace system. A second part of the answer is that after the model has been employed as a basis for identifying prime activities in a given development cycle, then the resulting development cycle design provides a framework which can be elaborated to any extent required to take into account the unalterable facts of the real world of system development. Whatever embellishment is added, however, the prime functions must remain if the resulting design is to be one which is capable of success. Or to put it another way, any successful development cycle can be shown to have as its framework an arrangement of prime activities which can be related to the "Go" model we have just discussed.

In this section of the report we will concern ourselves with some of the principles by which a selected prime development cycle design may be elaborated for the purpose of obtaining a design that will work in the real world of system development.

A real-world development cycle is complicated for good reasons; the twists and turns which it takes are not capricious. It will be useful now to consider some of the reasons underlying the complexity of a typical development cycle.

Just as it is appropriate to be concerned with the "goodness" of an operational system, and to define an act of measurement (the Quality Score formula) by which we may determine its goodness, so it is appropriate to be concerned with the "goodness" of a development cycle. We have been referring to the measure of goodness of a development cycle as Dev Q. We need now concern ourselves with the factors in Dev Q, because these factors identify the problems which one seeks to solve when he complicates or elaborates a prime development cycle design.

First, to be successful a development cycle must deliver an operational system with the target Quality specified by the customer in the Basic System Specification. Second, the operating costs of the delivered system must be within the limits set for operating costs in the Basic System Specification. To avoid future circumlocutions, we will refer to these two attributes of the delivered operational system taken together as Q/Op Cost.

If we assume that a development cycle will deliver an operational system with the target Q/Op Cost, then to measure the goodness of the development cycle we are likely next to turn to consideration of the time consumed in the development process. Occasionally, the calendar time at which the operational system is delivered out of a development cycle is not of consequence, but ordinarily, because of obsolescence, if for no other reason, there is concern with time. To be good, some development cycles must deliver their outputs within an interval of calendar time. Most often, however, when time is of consequence the outputs must be delivered " no later than" a given point in time.

The third common important factor in Dev Q is development resources; usually there is a limit to the resources that can be expended for development. Often it is possible to write a simple formula in which there is a maximum development cost beyond which Dev Q rapidly falls toward zero, but such that, in general, the lower the cost (in terms of resources consumed) the higher Dev Q becomes.

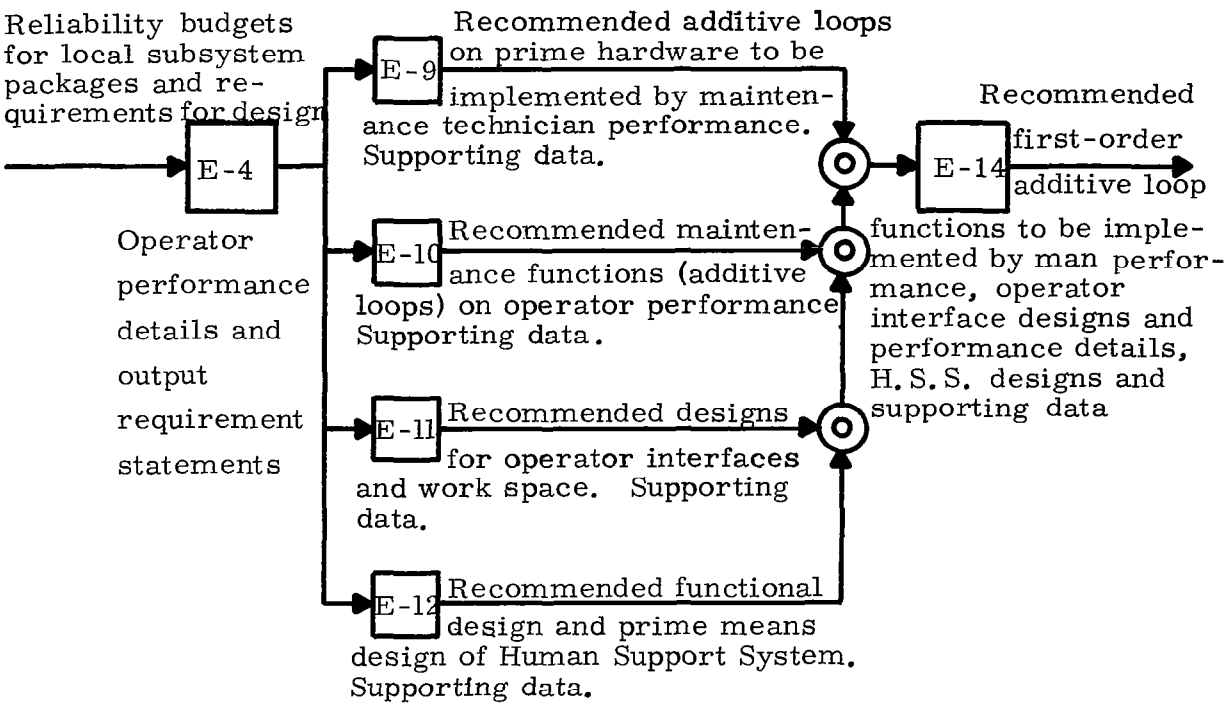
In this chapter we consider three factors in Dev Q: Q/Op Cost, time, and Dev Cost (where Dev Cost reflects all resources necessary to prosecute the development cycle).

We will turn first to consideration of Q/Op Cost. At the beginning of a development cycle, we are concerned with the probability that the development cycle will produce an operational system with the desired Q/Op Cost. At the end of a development cycle, the matter of probability has been decided and probability cannot enter into an after-the-fact evaluation of a development cycle. However, the problems of concern which have led to consideration of the development cycle model presented here are problems of prediction and control, not problems of after-the-fact evaluation. It is therefore appropriate that we concern ourselves with techniques for assuring that a development cycle will be highly likely to produce a system with acceptable Q/Op Cost. The techniques for achieving this objective are here called technical management. In a section below we will discuss a general principle by which the development cycle model may be elaborated to account for technical management.

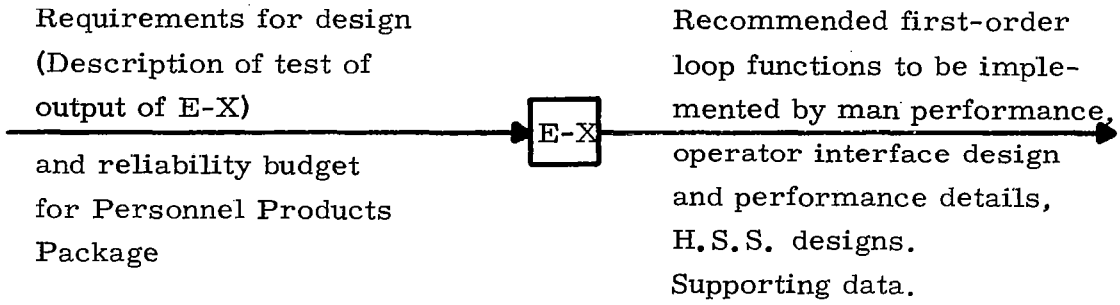
Following our discussion of technical management, we will turn to consideration of general management, which encompasses all of those activities carried out for the purpose of achieving a development cycle which delivers its output "on time" and " in the money".

All management activities taken together make up what might be called the management set. The activities in the management set are all concerned with the "goodness" of the development cycle. They are activities which are incapable by themselves of producing the output of the development cycle, but which may be used to extend a prime development cycle design in order to improve its Dev Q. Activities in the management set are needed in every real-world development cycle simply because the probability that a development cycle will proceed without error, on schedule and without cost overruns is very, very low indeed if the prime design alone is used.

Let us consider a typical set of activities in the model which is bracketed by a pair of activities at a more gross level of organization of the operational system under development. For example, let us consider the array shown below which is taken from the local segment side of the model for Function E. In the diagram below, Functions E-9, E-10, E-11, and E-12 are bracketed by activities E-4 and E-14. These activities are concerned with the personnel products package at the Function E level of design.

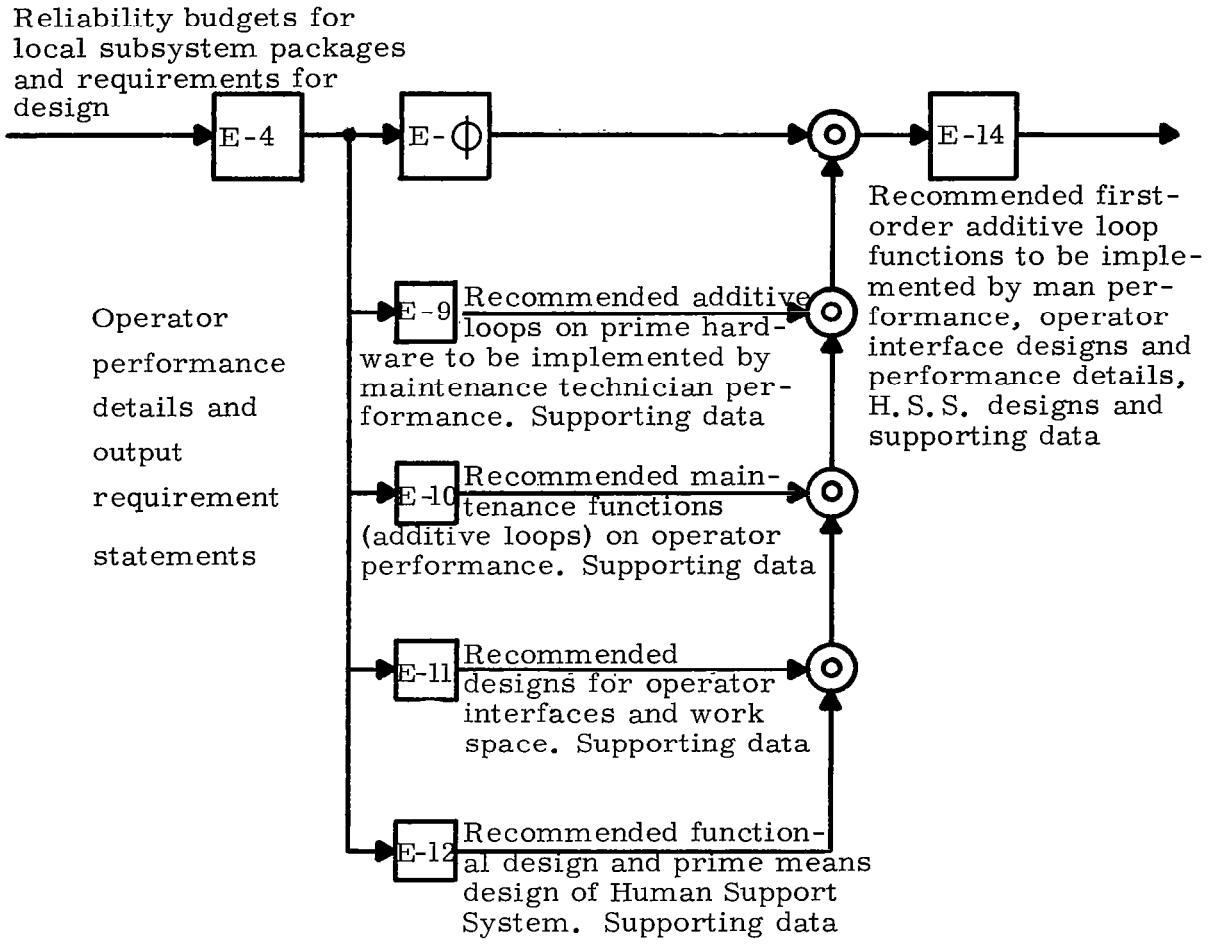


The activities (functions) in the above diagram might be added together and expressed as a single function as in the following diagram.



The implication of the above diagram is that Function E-X responds to a "test specification" from the segment level in the overall model, and that it delivers to the segment level its required end product plus data to demonstrate that the end product has passed the test specified earlier at the segment level. Returning now to the former diagram, it can be seen that activity E-14 must deliver the output state of Function E-X, but that it encompasses only the assembly of the end products of activities E-9, E-10, E-11, and E-12—it does not account for the development of these end products. In a "GO" model, it is perfectly reasonable to partition E-X in the manner shown above because we assume that the end products of activities E-9 through E-12 will not have any errors in them. However, in the real world of system development, the outputs of Functions E-9 and E-12 will almost certainly require some sort of mutual adjustment before they could be "fitted together" in activity E-14. If the output of one or more of these activities (E-9 through E-12) were very bad, the work might have to be done over again in activity E-14. This makes the work of activity E-14 a rather uncertain thing. It means, for example, that E-14 might be done in very little time and with minimum resources in the case of good inputs to it, or that it might take a considerable amount of time and a large commitment of resources if the inputs to it were very bad.

Now it may be observed that the characteristics of the inputs to activity E-14 are specified in the output of activity E-4. Thus, in the diagram there is a hiatus of concern at the personnel products package level between the output of activity E-4 and the input to activity E-14, and it is during this hiatus that things may go astray such that the input to activity E-14 becomes uncertain. Clearly, it would be desirable to ensure that the input to activity E-14 is always good, and to do this requires the insertion of a function which is concerned with the goodness of that input and which operates during the same period of time as Functions E-9 through E-12. In the figure below, we have inserted such a function, and we have labeled it a ϕ function.



The Φ Function

The Φ function in the diagram above is one which acts on the basis of the output of activity E-4 to control activities E-9 through E-12 such that their output states will satisfy the test specifications of activity E-4 when they are delivered. The Φ function is at the personnel package level. In order to accomplish its task, it must operate continuously to monitor activities E-9 through E-12, to test whether or not their outputs will be integratable, to foresee whether or not their outputs will meet the test specifications, and to take corrective action whenever it appears that some error will occur if corrective action is not taken.

We may take the Φ function discussed above as typical of a class of functions which might be employed at the system level, at the segment level, at the package level, and at lower levels throughout the course of system development in Functions D through H, to ensure that the delivered end product will be of satisfactory Q/Op Cost. We may call the set of all such Φ functions the technical management set. The technical management set is essentially an additive set. It employs monitoring and corrective actions within the course of system development to ensure a high probability that the desired development cycle output will be obtained.

General Management

The conduct of a development cycle requires resources and time, and if these are to be used within limitations then steps must be taken for their management. The management (or control) of time and resources is not easily separated into two parts, however, for it is frequently possible to trade off time in order to save resources and to employ extra resources in order to save time.[1]

---

[1] Not only is there an interaction in the management of time and resources, but there is also an interaction between time and resources management and technical management. We recognize this fact by referring to the management set. In order to simplify the presentation of the management discussion, however, it has been convenient to separate the management set into technical management and general management.

Under general management we must include activities which focus upon detecting and correcting trends in the course of development that would put the development cycle over the mark in terms of time or Dev Cost. Such activities are analagous in effect to the technical management activities discussed above. But in addition to these, under general management we must also include activities which are concerned with the design of the development cycle itself and activities which are concerned with seeing to it that the basic resources necessary to prosecute the development cycle are provided.

In introducing the task of designing the development cycle, we have come full circle, for the primary application of the model that is presented in this report is in such design. The model is, of course, intended to be useful only for setting the general course of design; it does not provide guidance for setting for the details of a development cycle design. The associated simple calculus for discrete systems provides a tool to enable design of a development cycle to whatever level of detail is necessary for management purposes.

In the design and control of a development cycle that will deliver its output within resource limitations, general management must provide and control the means by which the development cycle itself is implemented. Thus, there must be provided engineering personnel, tools and equipment needed in the design and development process, facilities, basic data services and the like. In order to avoid resource waste, general management must schedule resource use. By scheduling, management will avoid the need to duplicate resources by staggering usage and by employing "multipurpose" resources. System testing (such as would be necessary to develop supporting data in Function H) is an example of an activity requiring careful provisioning by management. In testing, it will often be true that a mock-up or simulator required for one purpose can easily be modified to serve another purpose if the scheduling of test activities is appropriately staggered, thus conserving development resources.

The scheduling of development cycles, and the provisioning of resources is often carried out by the use of a modeling technique called PERT (ref. 16). PERT models are well-suited to the task of determining and displaying the relationships among activities as a master "timetable." The PERT technique can also be used to assist in the determination of the development cycle resources that are required. When a development cycle design is carried out in terms of the function concepts employed in this report, the resulting design can readily be translated into a PERT model.[1] Inasmuch as a PERT model is a superior vehicle for designing and implementing general management of time and resources, it will in general be best if the translation is made so that PERT may be used for these purposes. PERT, on the other hand, is an inferior vehicle for first laying out the design of the prime development cycle and for determining technical management requirements.

In sum, the management set includes technical and general management at all levels— at the system level, at the segment level, at the package level, and on down to the level of the smallest working group concerned with a definable subpackage of the system. Both technical and general management are necessary to achieve a high probability of success of a development cycle. The model and the modeling techniques presented in this report are useful for the purpose of designing the prime development cycle and for determining the technical management activities that are needed to insure the quality of the operational system under development. The modeling technique that is presented is readily translated into a PERT model. Such models are best suited to aid in the design and implementation of general management concerned with time and resource use to the end of achieving a satisfactory Dev Q.

---

[1] It has been shown that a model in terms of this function notation is also readily translated into a probability model. (See Part B of this report.)

Serendipity Associates
Chatsworth, California, October, 1966.

APPENDIX A
METHOD OF DEVELOPING THE MODEL

Any model that is represented as being useful for predicting and for controlling the course of a development cycle is bound to give rise to the question, "Is it good?" To answer this question directly would require that it be employed to predict and control several different aerospace system development cycles in order that the quality (Dev. Q) of these development cycles might be compared with the quality of other typical development cycles. It is not likely nor even advisable that such a test be conducted. To conduct such a test would be much too costly, too hazardous, and even too unlikely of success to be justifiable. Success in demonstrating the utility of the model would be unlikely for a number of reasons. First, the model is not an attempt to prescribe how a development cycle should be conducted; rather, it is an attempt to make explicit the implicit logic which underlies our current method of developing aerospace systems. Second, although the model may have some utility simply because explicit guidance is more certain than implicit guidance, to compare it at this stage of development with an "unguided" development cycle may be like comparing the first steam engine to the horse. Third, there are alternative indirect tests that are less certain but less costly by which this model may be assessed.

At least three alternative ways of evaluating the model should be considered. One way is to examine the logic by which the model has been constructed. The body of this report attempts to set forth that logic in a public manner so that it may be examined, criticized, revised, and improved, or perhaps rejected. A second method of indirect evaluation is to test the model against known facts about specific aspects of system development. This is done in part in Report II of this series. In this report (see p. iii) man-related activities in the model are further described and developed. The model requires a better examination than it has been given in Report II, however. It should be tested for its capability to encompass the facts of hardware development more comprehensively than has been possible to date. A third way to evaluate the model is to examine the manner in which it was developed. In this appendix, the method by which the model was developed will be described so that it may be examined.

The model was generated in two stages. In the first stage, three independent alternative approaches to development of the model were undertaken with the expectation that the three alternatives would be melded into one model. When this approach turned out not to be successful, one of the three models was selected as a basic one and in Stage 2 this selected model was elaborated and refined until the model presented in this report emerged. As a matter of record and in order to identify major source materials and the manner in which they were employed, both stages will be described in more detail below.

Stage 1

In this stage, three parallel efforts at model development were undertaken. For convenience, we will name these A, B, and C. Inasmuch as it was intended to weave all three models together after they had been completed, the three independent efforts were carried out under certain common restraints and guidance. Thus, all were guided by several working documents, one of which defined the basic terms and concepts to be employed. This working document evolved eventually into chapter II of this report, Conventions and Assumptions. A second working document prescribed the symbolic method of modeling; it is represented in the report series by Report V, A Simple Calculus for Discrete Systems. A third working document entitled, Rules for Preparing Development System Models was prepared to guide the three modeling efforts. This document identified the input and output boundaries of a development cycle and suggested that all models employ the basic three-phase breakout that is given in this report. This was suggested (but not required) in order to facilitate comparisons of the three independently prepared models. The rules also suggested the level of detail to which the model should be carried and called for focus of attention upon activities concerned with personnel products. The rules suggested that general management functions should not be incorporated in the models and that the models should be "GO" models. A fourth working document was entitled Outputs of Typical NASA Development Systems. This document

identified the typical categories of outputs that would be found upon examining any aerospace system development cycle, and thus it identified the minimum list of outputs which each of the models should account for. The classes of outputs identified in this working paper were derived by analysis of existing aircraft systems and existing space systems. The categories identified were relatively gross with most detail being given to personnel products.

Taken together all of the working documents described above provided the context within which development cycle models A, B, and C were produced. Model C was prepared without any additional guidance. Model C was developed on the basis of previous efforts to describe the man-machine system development process such as the description given in ref. 27 . It was developed with aircraft development cycles as the point of departure; specific consideration was given to the problems of development of the SST. Models A and B were developed under additional ground rules not employed in the case of model C. The developers of models A and B were obliged to account for a specific list of categories of end products necessary to configure a manned Mars exploration system. On the basis of a review of Mars mission studies (ref. 30, 10, 19, 26, 31, and 24), a list of typical operational system means was developed and set forth in a working document as a target for models A and B. A rough approximation of a Basic System Specification for a Mars system was also prepared as a working document to guide the efforts to prepare models A and B.

Models A and B differed with respect to the sources of information employed. Model A was an amalgamation of existing documented development cycle "models." The existing models employed were:

a. The Air Force 375 series approach (ref. 1, 2, 3, 4, 5, and 6);

b. Guidance for the development of United States Navy systems (ref. 14);

c. The model given in Introduction to Design, a recent book on the topic of system development (ref. 8);

d. Information given in another book which is an important source of system engineering information, System Engineering by Goode and Machol (ref. 21).

A "box and arrow" representation of the model described in each of these sources was prepared in order that all of the source models might be available in the same format. The model which contained the greatest amount of detail was the one which represented the Air Force "375" approach. Model A was prepared as a description of the development process for a manned Mars system based upon these four model sources.

Model B was also designed to account for the delivery and installation of a manned Mars system. It was based primarily upon a development cycle logic set forth in an earlier unpublished work.[1]

After the three models had been prepared independently, an attempt was made to amalgamate them. This attempt did not bear fruit; no rationale could be found by which to accomplish the process of amalgamation. It was felt that the expert judgment of a small number of people would not provide either a public or justifiable basis for setting forth any compromise model. Therefore an alternative approach was sought, giving rise to Stage 2 in the generation of the development cycle model.

Stage 2

Of the three models prepared in Stage 1, model B was supported by the most complete rationale. The approach in Stage 2 was to develop this model further on the basis of logic and the concepts contained in the simple calculus for discrete systems and then to crosscheck and correct the resulting model. Model B was thus further elaborated; it was then given three checks, each of which resulted in some modification.

---

[1] Wulff, J. J.; Inaba, K.; and Pool, E. T.; A Guide for the Development of Training Materials and Personnel Products for Man-Machine Systems. Psychological Research Associates, Inc., Arlington, Virginia, 1959.

After a reasonable review of the current literature that is germane to the conduct of development cycles for complex systems, and after a review of model B by two consultants,[1] model B was improved and given its first check. This was done by comparing model B with models A and C for the purpose of determining gaps or inconsistencies in logic which might be revealed by this comparison. Model B was also checked against the symbolic representations of the documented models in the literature that were employed as the basis for generating model A. These checks resulted in a number of changes in model B.

The second check that was given the elaborated model B was focused specifically upon determining its completeness with respect to coverage of development cycle activities related to personnel products. A number of books and reports (ref. 15, 7, 1, 2, 3, 4, 5, 6, 18, 20, 17, 29, 32, 9, 12, 13, 23, 25, 14 and 28) were reviewed for the purpose of preparing a card file in which each card named a development cycle activity related to personnel products. The resulting card file was over 600 cards in length and contained many redundancies. It was tested for its completeness by a matrix method and the gaps detected in the card file were filled by further reference to the literature and on the basis of the experience of human factors experts. The expanded model B was then checked against the card file, card by card, in order to determine whether or not there were cards which identified important development cycle activities not encompassed by the model. As a result of this check a small number of changes were made in the model.

The final check on the model was accomplished in the preparation of Report II of this series. This report contains detailed consideration of each of the man-related activities identified in the development cycle model. These activity descriptions were prepared by human factors experts capable

[1] Model B was reviewed by Professor Warren E. Wilson, Chairman of the Engineering Department, Harvey Mudd College and author of Engineering System Concepts (ref. 33). The model was also reviewed by Dr. Elliot Axleband, a control system engineer. Both of these experts made important contributions to the improvement of model B.

of criticizing the model on the basis of their specialized experience in system development programs. The result of this review of the man-related activities in a development cycle model was to suggest several minor modifications.

The model, as presented in this report, reflects the changes suggested by all three of the checks described above.

REFERENCES

1. Air Force Systems Command: Configuration Management During Definition and Acquisition Phases. AFSC, Systems Management, AFSCM 375-1, 1 June 1964.

2. Air Force Systems Command: System Program Management Surveys and Industrial Management Assistance Surveys. AFSC, Systems Management, AFSCM 375-2, 25 June 1963.

3. Air Force Systems Command: System Program Office Manual. AFSC, Systems Management, AFSCM 375-3, 15 June 1964.

4. Air Force Systems Command: Systems Program Management Manual. AFSC, Systems Management, AFSCM 375-4, 16 March 1964.

5. Air Force Systems Command: System Engineering Management Procedures. AFSC, Systems Management, AFSCM 375-5, 14 December 1964.

6. Air Force Systems Command: Development Engineering (DE). AFSC, Systems Management, AFSCM 375-6, 14 August 1964.

7. Air Force Systems Command: Handbook of Instructions for Aerospace Personnel Subsystem Designers. AFSC Manual Nr 80-3, HQ, AFSC, Andrews AFB, Washington, D.C., April 1965.

8. Asimow, M.: Introduction to Design. Prentice-Hall, 1962.

9. Bennett, Edward; Degan, James; and Spiegel, Joseph: Human Factors in Technology. McGraw-Hill Book Co., Inc., 1963.

10. Boeing Company, The: Study of Interplanetary Mission Support Requirements. Prepared for Manned Spacecraft Center, Contract No. NAS9-3441, Final Tech. Rpt. D2-23588-5, Aero-Space Div., Seattle, Washington, May 1965.

11. Bridgeman, D. W.: The Logic of Modern Physics. New York, 1927.

12. Brown, K.; and Weiser, P., eds.: Ground Support Systems for Missiles and Space Vehicles. McGraw-Hill (New York), 1961.

13. Brown, Kenneth, et al.: Space Logistics Engineering. John Wiley & Sons, Inc., 1962.

14. Bureau of Naval Personnel: The CAPRI System for Naval Manpower Planning and Control. Preliminary Report, Personnel Research Div., Dept. of the Navy, 1963.

15. Chaillet, R. F.: Human Factors Engineering Requirements for the Development of U.S. Army Materiel. Human Engineering Lab., HEL Standard S-4-65, 1965.

16. Department of Defense; and National Aeronautics and Space Administration: DOD and NASA Guide PERT COST. Joint services, Office of the Secretary of Defense and National Aeronautics and Space Administration, June 1962.

17. Fogel, Lawrence J.: Biotechnology: Concepts and Applications. Prentice-Hall, Inc., 1963.

18. Gagne, Robert M., ed.: Human Functions in Systems. Psychological Principles in System Development, Holt, Rinehart and Winston, 1962.

19. General Dynamics Corporation: A Study of Mission Requirements for Manned Mars and Venus Exploration. Prepared for NASA, George C. Marshall Space Flight Center, Contract No. NAS8-11318, General Dynamics/Fort Worth, Report FZM-4366, 30 May 1965.

    Volume 1 Research and Technology Implications Report
    Volume 3 Technical Report

20. Gerathewohl, Siegfried J.: Principles of Bioastronautics. Prentice-Hall, Inc., 1963.

21. Goode, Harry H.; and Machol, Robert E.: System Engineering: An Introduction to the Design of Large-Scale Systems. McGraw-Hill Book Co., Inc., (New York), 1957.

22. Gosling, W.: The Design of Engineering Systems. John Wiley & Sons, Inc., (New York), 1962.

23. Hanes, Lewis F.; Ritchie, Malcolm L.; and Kerns, III, John H.: A Study of Time-Based Methods of Analysis in Cockpit Design. ASD-TDR-63-289 (DDC AD 408782), Flight Control Lab., Aeronautical Systems Div., Air Force Systems Command, WPAFB, May 1963.

24. Lockheed Missiles and Space Company: Manned Interplanetary Missions. Follow-on study final report, vols. 1, 2, and 3, Prepared for George C. Marshall Space Flight Center, Contract NAS8-5024.

25. Meister, D.; and Rabideau, G. F.: Human Factors Evaluation in System Development. John Wiley & Sons, Inc., 1965.

26. North American Aviation, Inc.: Manned Mars Landing and Return Mission Study. Final Report, vol. 3, Rept. No. SID 64-619-3, (Contract No. NAS2-1408), Space and Information Systems Div., (Los Angeles), April 1964.

27. Price, Harold E.; Smith, Ewart E.; and Behan, R. A.: Utilization of Acceptance Data in a Descriptive Model for Determining Man's Role in a System. NASA CR-95, 1964.

28. Pulscak, M. W., et al.: The CAPRI System for Naval Personnel Program Management. Operations Research, Report Nos. ND 65-28 and 29, Tech. Report 322 (Contract Nonr 3949(00)), 1965.

Volume I Description and Operation
Volume II Data Processing Manual

29. Schaefer, Karl E., ed.: Bioastronautics. Macmillan Company, (New York), 1964.

30. Serendipity Associates: Final Report of a Study of Crew Functions and Vehicle Habitability Requirements for Long Duration Manned Space Flight. Ames Research Center, NASA, Contract NAS2-2419, August 1965.

Volume I Summary Report
Volume II Technical Program
Volume III Technical Appendices

31. Thompson Ramo Wooldridge, Inc.: Manned Mars Landing and Return Mission. Vols. 1 and 2, No. 8572-6011-RU-00, (Contract No. NAS2-1409).

32. Walton, T. F.: Technical Data Requirements for Systems Engineering and Support. Prentice-Hall, 1965.

33. Wilson, W. E.: Concepts of Engineering System Design. McGraw-Hill, 1965.

PART B
A SIMPLE CALCULUS FOR DISCRETE SYSTEMS

CONTENTS

I. THE NEED FOR A CALCULUS

The complex utility systems and weapon systems that are built out of public resources are basically problem solving systems; society buys them in the hope that they will serve to solve problems that broadly affect society. In recent times such systems have been developed at an accelerating rate. To some extent each new system builds upon the systems that have been developed in the past, and, in this manner, the overall size and complexity of systems tends to grow. Although this growth in size and complexity has made these new systems more difficult to produce, society has nevertheless produced many of them when urgent pressures have been broadly recognized. Thus, in recent years society has been able to solve problems that were thought impossible of solution a few years ago. The importance to society of the new complex systems is great and there are strong pressures to continue to solve those problems that are shifted into the realm of solvability by advancing technology.

With increasing complexity and with increasing importance of the problems encountered, there has also been a trend toward increasing cost of systems. Thus, in our time, new systems to serve society, such as waste management systems, power supply systems, and transportation systems require for their development such a significant proportion of our total resources that we cannot undertake them all at once, even though all are clearly within the scope of technology to build. Given as opposing factors high cost in terms of resources needed for development and great importance in achieving success, there is need for capability to predict, to design, and to control development processes for the complex systems needed, so that our resources can be used most effectively to solve as many problems as possible.

In recognition of the importance of control over the development process, in recent years there has been increasing use of one tool that is useful for

this purpose, the Program Evaluation and Review Technique called PERT. PERT was developed specifically to help solve the problem of gaining control of the development process and it does provide a partial answer to the need. However, the successful use of PERT techniques for the control of a given development cycle depends upon having an adequate description of the development process to be controlled. Given an adequate description, PERT techniques can be employed to redescribe the process in terms of resource requirements, time requirements, and contingencies. But without any description of the steps in the development process to start with, PERT is of no use. By the same token, a good PERT description cannot offset a bad process description upon which it is based. To date it appears that there is no generally available method for generating an adequate description of a development cycle so that a PERT description may be generated in turn and used to full advantage. There is a need for such a descriptive method.

There have been attempts to describe or model the process of complex system development. The most significant undertaking has been sponsored by the Air Force. With the support of the Department of Defense (ref. 3), the Air Force has prepared an horrendously detailed description of the process by which the systems built under its aegis should be developed (ref. 1). The Air Force documentation, however, does not lend itself to adaptation for solving the development cycle problem in general. It is tailored specifically for the management conventions and hierarchical relationships of the Air Force. It presents a model for system development in great detail, but the model is not one from which general principles may be extracted, nor is it one that is amenable to evolution by means of rigorous public discussion. Several authors writing in the general area of system engineering have recently presented models of what the system development process is like (ref. 2, 6, 5); none of these contains sufficient detail nor adequate rationale for it to be useful for solving the problem of gaining control of the system development process.

Although existing documented descriptions of the development process are not adequate to enable the prediction, design, and control of development cycles for complex systems, they all demonstrate that the business of designing

a development cycle is essentially that of finding a defensible strategy for the sequence and relationships of events that must take place in the course of developing a complex system. In order to be able to talk about development cycle strategies without ambiguity, and in order to promote the comparison of alternatives in the course of evolving good strategies, we need a special language; specifically, there is need for a language whose terms and concepts are public and precise and whose symbology is well defined so that there can be an exchange of precise ideas among the specialists interested in the development process. Given such a language, there would be a good basis for communicating and improving development cycle models which exhibit useful strategies.

The objective of this paper is to present a language which satisfies the needs outlined above.

The vehicle for talking about development systems that is presented here was generated within certain ground rules. A basic rule was, of course, that the language be useful for talking about development cycles. Another ground rule was that the language be presented as a calculus according to the conventions of mathematics in order to take advantage of the established methods of the mathematical community as a way of providing for the orderly improvement of the language (ref. 7). Yet another ground rule was that the calculus should articulate with PERT and with probability calculus, such that it would permit building models of development cycles which could be translated into probability equations (models) on the one hand, or into PERT models on the other (ref. 4). This ground rule was compatible with our objective that the language make it possible to utilize computers for testing and manipulating detailed development cycle models which might result from the use of the language. Finally, we hoped to provide a language rich enough to enable the evolution and elaboration of relatively complex models of the system development process, should such elaboration prove to be necessary and fruitful.

What follows then, is the presentation of a simple calculus which is a language for talking about development cycles. It is called a simple calculus

for discrete systems, because we believe that any development cycle may usefully be treated as a discrete system.[1] In this manner, we have avoided the complexity which would have been necessary had we chosen to attempt the development of a calculus for systems whose individual outputs must be described over an interval of time, or whose outputs are distributed over time. Only the test of application will reveal whether or not this was a good decision. Following the presentation of the calculus in the next section, there is a brief discussion which attempts to provide a partial justification for the specific coinage and syntax chosen for the calculus. The method of justification is to introduce the reader to the use of the calculus for describing development cycles.

II. THE CALCULUS

We begin the presentation of the calculus with a discussion and definition of the key concept, State.

State

In the definition of state, we shall employ the intuitive concept, "public method of measure." By public method of measure we mean a set of instructions which is available to a target population of people, and which, when used by members of this population, is capable of reliably guiding their actions in obtaining information about the real world. We call the information obtained (that is, the result of using the method of measure) a "symbolic statement" (e.g., 26 grams). For our purposes, a symbolic statement is a sequence of symbols from an appropriate underlying alphabet.

---

[1] A discrete system is one whose operation can satisfactorily be described as a finite sequence of events moving forward in time and whose terminal output state is fully described at a point in time after which no further events occur. Such a system must be one whose condition at any point of time can satisfactorily be described by stopping the clock and by identifying the complete condition of the system at that point in time. (Output state is precisely defined in the following section of this paper.)

For the purposes of the people who would obtain a symbolic statement by measurement, it would hopefully convey some information about the real world, which would be of use to them. We shall assume that there is a basic encyclopedia, $\mathcal{E}$, of methods of measure which is publicly available, and from which precisely defined methods of measure may be drawn.

We call the use of a public method of measure "an act of measurement." We assume that an act of measurement occurs at a particular point in time. In fact, measurements are not made at a point in time, but in most applications of the calculus there is no penalty for pretending that a symbolic statement is associated with a point in time, and to do so avoids need for undue complexity in the calculus.

We are now prepared to begin a rigorous definition of state. Roughly speaking, we want our definition of state to carry the idea that a state is the symbolic statement resulting from an act of measurement. As it turns out, as soon as we get state pinned down to this idea, we will want to expand the notion of state to include other ideas as well. Therefore, let us call this preliminary notion an atomic state; we will save the word state for the expanded notion which will come a little later on. Thus, we define atomic state rigorously as follows:

> Definition: — An atomic state is an ordered triple (S, M, t), where S is a symbolic statement, M is a public method of measure taken from the basic encyclopedia $\mathcal{E}$, and t is a symbol (frequently a real number). We interpret this triple as follows: S is the symbolic statement which results from the use, at time t, of the public method of measure M.

As already stated, for our purposes a symbolic statement is a sequence of symbols from an appropriate underlying alphabet. Likewise, M may be thought of as a sequence of symbols; namely, that sequence of symbols which makes up the set of instructions of which M is composed.

Suppose that M_1 and M_2 are public methods of measure. Then a combined method of measure might for didactic purposes be thought of as devised by tagging the set of instructions for M_2 at the end of the set of instructions for M_1. Then as one finished complying with the instructions

for M_1, it would still remain to comply with the instructions for M_2. If people from the target population may reliably use M_1 and also reliably use M_2, they may reliably use the combined public method of measure (denoted by M_1M_2). If S_1 and S_2 are the symbolic statements resulting from the use of M_1 and M_2 at some common instant in time, then we denote by S_1S_2 the symbolic statement resulting from the use of M_1M_2 at that same instant in time. We may now rigorously define the expanded notion of state which we need.

Definition: — We define state recursively in terms of atomic state as follows:

1. Every atomic state is a state.
2. If (S_1, M_1, t_1) is a state, and (S_2, M_2, t_2) is a different state but with $t_1 = t_2$, then (S_1S_2, M_1M_2, t_1) is a state.

Thus, our general notion of state, is that a state is something which may be composed of atomic states or other states. Notice that the recursive definition above permits states of the following structure: $(S_1S_2 \ldots S_n, M_1M_2 \ldots M_n, t)$. Therefore, we may think of a state as being composed of many states. That is, the state $(S_1S_2 \ldots S_n, M_1M_2 \ldots M_n, t)$ may be thought of as being composed of (S_1, M_1, t), and (S_2, M_2, t), and,... and (S_n, M_n, t). We shall call the set of states $\{(S_1, M_1, t), \ldots (S_n, M_n, t)\}$ a subdivision of the state $(S_1S_2 \ldots S_n, M_1M_2 \ldots M_n, t)$. Notice that a given state may have many subdivisions. Thus if we let $S' = S_2 \ldots S_n$ and $M' = M_2 \ldots M_n$, then $\{(S_1, M_1, t), (S', M', t)\}$ is also a subdivision of $(S_1S_2 \ldots S_n, M_1M_2 \ldots M_n, t)$.

Instead of writing a triple each time we wish to refer to a particular state, we shall often use a single lower case letter (sometimes with a subscript) to denote a state. Thus, for example (S, M, t) might be denoted by b, and (S_1, M_1, t_1) might be denoted by b_1. In addition, we shall sometimes denote the state which results from combining $b_1, \ldots b_n$ as $\{b_1, \ldots b_n\}$. (Notice that $b_1, \ldots b_n$ must all have the same time t associated with them, else they cannot legally be combined into a single state.) Furthermore, if b is composed of the states $b_1, \ldots b_n$, then of each b_i we shall say that b_i is an element of b. Finally, we shall sometimes use the

convention of referring to the time t associated with the state (S, M, t) as "the time at which the state occurs."

Primitive Functions

We do not use the word function as it is used in the world of mathematics. Rather, our use of the word derives from its everyday use in systems analysis. We define primitive function precisely as follows:

> **Definition: — A primitive function is an ordered pair of states (a, b) such that if t_a is the time associated with a, and t_b is the time associated with b, then $t_a < t_b$ (read, "t_a earlier than t_b").**

The first coordinate in the pair is called the input state, and the second coordinate is called the output state.

Isomorphic Primitive Functions

Let (a, b) be a primitive function and suppose a and b are subdivided as follows: $a = \{a_1, \ldots a_n\}$ and $b = \{b_1, b_2\}$. Eventually we would like to be able to make sense of such questions as:

1. "What is the probability that b_1 and b_2 occur, given that all of $a_1, \ldots a_n$ occur?"
2. "What is the probability that b_1 or b_2 or both occur, given that all of $a_1, \ldots a_n$ occur?"
3. "What is the probability that b_1 occurs given that a_7 or a_8 or both occur?"

To answer such questions as these, we must define an appropriate sample space over which to compute probabilities. This sample space will be defined in the following section, and will consist of a collection of primitive functions which are in some way "similar." This notion of similarity is contained in the following definition of isomorphic primitive functions.

Definition: — Let $F^1 = (a^1, b^1)$ and $F^2 = (a^2, b^2)$ be primitive functions, which are subdivided as follows: $a^i = (a^i_1, \ldots a^i_n)$ and $b^i = (b^i_1, \ldots b^i_m)$ where $i = 1, 2$. Then F^1 and F^2 are isomorphic if:

1. $t_{b^1} - t_{a^1} = t_{b^2} - t_{a^2}$, where t_{b^i} is the time associated with b^i, and t_{a^i} is the time associated with a^i, for $i = 1, 2$.
2. For each a^1_k, the method of measure for a^1_k is the same as the method of measure for a^2_k.
3. For each b^1_k, the method of measure for b^1_k is the same as the method of measure for b^2_k.

Roughly speaking, for F_1 and F_2 to be isomorphic, there must be a subdivision of the input state of F_1 and a subdivision of the input state of F_2, which both have the same number of substates. (Likewise there must be subdivisions of the output states of F_1 and F_2 with the same number of substates in them.) In addition, the time difference between the input and output of F_1 must be equal to the time difference between the input and output of F_2. Furthermore, the method of measure in each substate in the input subdivision of F_1, must be the same as the method of measure for the corresponding substate in the input subdivision of F_2. (A similar condition holds for the output subdivisions of F_1 and F_2.)

Probability Tables for Primitive Functions

Let F be a primitive function (a, b), where a and b are subdivided in some desired manner: $a = \{a_1, \ldots a_n\}$, $b = \{b_1, \ldots b_m\}$. With this primitive function F and its subdivisions, we associate a class of sample spaces. Each sample space X in this class is a finite set of primitive functions $\{F_1, \ldots F_n\}$, each of which is isomorphic to F. In addition we assume that X contains F. The probability measure, P, on X is defined by the number of elements in the subsets of X. Thus if Y is a subset of X, then $P(Y) = \frac{m}{n}$, where m = number of primitive functions in Y, and n = total number of primitive functions in X. Notice that P(X) = 1.

Example: — Let F be the primitive function $\big((S, M, t), (S', M', t')\big)$. Then the following set of primitive functions is a sample space for F:

F_1 (=F): $\big((S, M, t), (S', M', t')\big)$

F_2: $\big((S_1, M, t+\Delta), (S_1', M', t'+\Delta)\big)$

F_3: $\big((S, M, t+2\Delta), (S_2', M', t'+2\Delta)\big)$

F_4: $\big((S, M, t+3\Delta), (S', M', t'+3\Delta)\big)$

Where $S \neq S_1$, and S', S_1', and S_2' are all distinct.

In this example, neither the input nor the output of F is subdivided further than one state apiece, whereas in the general case the subdivision may be quite extensive in each state. Notice however that the input and output states of the other primitive functions F_2, F_3, and F_4 in the sample space are subdivided to the same extent that the input and output of F is. Thus in this example the inputs and outputs of F_2, F_3, and F_4 each have only one state in them, in consonance with F. Notice also that the method of measurement in each input state of F_2, F_3, and F_4 is the same method of measurement as in the input state of F. Likewise the method of measurement in each output state of F_2, F_3, and F_4 is the same as the method of measurement in the output state of F. Finally, observe that the time differences between the input and output states of the primitive functions in the sample space X are:

time difference $= t' - t$ for F,

time difference $= t' + \Delta - (t + \Delta) = t' - t$ for F_2,

time difference $= t' + 2\Delta - (t + 2\Delta) = t' - t$ for F_3, and

time difference $= t' + 3\Delta - (t + 3\Delta) = t' - t$ for F_4.

Thus all the time differences are identical. In summary, then, we have shown that all the conditions for isomorphism exist between F, F_2, F_3, and F_4, and hence X is indeed a sample space for F. Actually one would probably want a sample space to contain a very large number of primitive functions in it, rather than a mere four primitive functions as in this example. Therefore in practice, a sample space so simple as this one would not be used.

Special Notation: — If a is any state, then let S_a be the generic symbol for the symbolic statement within a. Thus, for example, if b_i is a state, then S_{b_i} denotes the symbolic statement in b_i.

Suppose F is subdivided[1] in some way, and let a be any substate in F (in either the input or the output of F). Let X be a sample space for F. Then we shall define a special subset X_a of S as follows. Let F' be any primitive function in X. Since F and F' must be isomorphic, there is a state a' in F' which corresponds to the state a in F. Then we shall let F' be a member of X_a if and only if $S_a = S_{a'}$. Evidently then F itself is a member of X_a. We call the set X_a the occurrence set for the state a. For each F' in X_a, we shall say a occurs in F'.

In the example above, the occurrence set for the state (S, M, t) is the set of functions $\{F_1, F_3, F_4\}$. Or, alternatively, we may say (S, M, t) occurs in F_1, F_3, and F_4! Notice that F_2 is not included because the symbolic statement in F_2 which corresponds to S, is S_1, and we assumed in the example that $S_1 \neq S$.

For any sets W and Z, $W \cap Z$ represents the set theoretic intersection of W and Z, $W \cup Z$ represents the set theoretic union, and $-Z$ represents the set theoretic complement of Z. We shall define a collection of subsets, Γ, of X as follows:

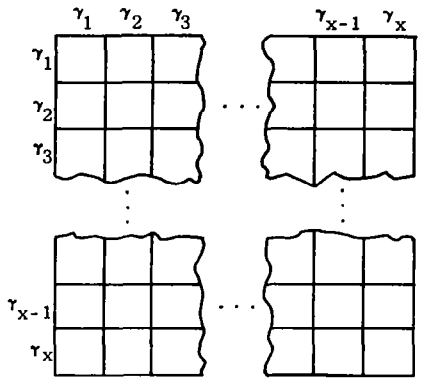
1. Γ contains X_c, for every state c in the subdivision of the input and output states of S.
2. If X_1 and X_2 are in Γ, then $X_1 \cap X_2$, $X_1 \cup X_2$, $-X_1$, and $-X_2$ are all in Γ.
3. Γ contains X.

Thus Γ is the closure under the operations $\cap$, $\cup$, and $-$, of the sets $X_{a_1}, \ldots X_{a_n}, X_{b_1}, \ldots X_{b_m}$. Let us assume that $\gamma_1, \gamma_2, \gamma_3, \ldots \gamma_x$,

[1] When we say that a function is subdivided, we employ the same idea as in the subdivision of states. Thus, a subdivided function is one whose input states and output states are expressed in subdivided form.

are the sets in Γ. Thus γ_1 might be X_{a_1}, and γ_{26} might be $X_{a_1} \cup \left(X_{a_3} \cap (-X_{b_2}) \right)$.

Definition: — We define the probability table for the primitive function F to be a chart:



The labels along the top of the chart correspond to the sets in Γ. The same labels are used along the side of the chart. The entry[1] in the chart at the intersection of row γ_i with column γ_j is the ratio: $\frac{\#(\gamma_j \cap \gamma_i)}{\#(\gamma_i)}$ (see footnote 2 below). Thus this entry is the probability of γ_j given γ_i! The unconditional probability of γ_j (i.e., $P(\gamma_j)$), is the probability of γ_j given X which is the ratio: $\frac{\#(\gamma_j \cap X)}{\#(X)}$ or, since X contains γ_j the ratio: $\frac{\#(\gamma_j)}{\#(X)}$. This unconditional probability of γ_j also appears in some entry in the chart, since X is in Γ and therefore X corresponds to one of the labels. (Notice that all the probabilities along the main diagonal of the chart are 1, since for every γ_j, the probability of γ_j given γ_j is simply 1.)

Now we can answer the sort of question that was posed in the preceding section. In that section, we considered a primitive function (a, b), with subdivisions $a = \{a_1, \ldots a_n\}$, and $b = \{b_1, b_2\}$. The problem posed

---

[1] It is unnecessary to prescribe every entry in the probability table. Using $P(A \cup B) = P(A) + P(B) - P(A \cap B)$ and its generalizations all entries may be computed given relatively few.

[2] If γ is any finite set, then $\#(\gamma)$ means "the number of elements in γ."

there was to make sense of such questions as:

1. "What is the probability that both b_1 and b_2 occur, given that all of $a_1, \ldots a_n$ occur?"

2. "What is the probability that b_1 or b_2 or both occur, given that all of $a_1, \ldots a_n$ occur?"

3. "What is the probability that b_1 occurs given only that a_7 or a_8 or both occur?"

To answer these questions we first construct a sample space X for (a, b). Then the required probability for question number 1 is interpreted to be:

$$P = \frac{\#\left[(X_{b_1} \cap X_{b_2}) \cap (X_{a_1} \cap \ldots \cap X_{a_n})\right]}{\#\left[X_{a_1} \cap \ldots \cap X_{a_n}\right]}$$

In this expression[1] for P, the set $X_{b_1} \cap X_{b_2}$ is the intersection of the set of primitive functions in which b_1 occurs with the set of primitive functions in which b_2 occurs. That is, $X_{b_1} \cap X_{b_2}$ is the set of primitive functions in which <u>both</u> b_1 and b_2 occur. Likewise, $X_{a_1} \cap \ldots \cap X_{a_n}$ is the set of primitive functions in which <u>all</u> of $a_1, \ldots a_n$ occur. Finally, $(X_{b_1} \cap X_{b_2}) \cap (X_{a_1} \cap \ldots \cap X_{a_n})$ is the set of primitive functions in which b_1 <u>and</u> b_2 <u>and all</u> of $a_1, \ldots a_n$ occur. Thus, P is the number of primitive functions (in $X_{a_1} \cap \ldots \cap X_{a_n}$) <u>in which</u> $\underline{b_1}$ <u>and</u> $\underline{b_2}$ occur, divided by the total number of primitive functions in $X_{a_1} \cap \ldots \cap X_{a_n}$. This then is how we define the probability that b_1 and b_2 occur given $a_1, \ldots a_n$.

---

[1] Notice that each set which occurs in this expression corresponds to one of the γ_i's in Γ, by the way Γ was defined.

The required probability in question number 2 is interpreted to be:

$$P = \frac{\#\left[(X_{b_1} \cup X_{b_2}) \cap (X_{a_1} \cap \ \ldots \cap X_{a_n})\right]}{\#\left[X_{a_1} \cap \ \ldots \cap X_{a_n}\right]}$$

Finally, the required probability in question number 3 is interpreted to be:

$$P = \frac{\#\left[X_{b_1} \cap (X_{a_7} \cup X_{a_8})\right]}{\#\left[X_{a_7} \cup X_{a_8}\right]}$$

Special Notation: — For a primitive function with even a moderate number of states in a particular subdivision, the construction of a complete probability table would be wholly unfeasible. There are simply too many entries in the chart to actually fill them all in. Ordinarily, just a part of the probability table is filled in. What results, in the working situation, is a "short" probability table with only those entries of particular importance in the given situation being filled in. Perhaps the shortest, and also the most commonly used probability table is the following. Let F = (a, b) be a primitive function with subdivisions $a = \{a_1, \ldots a_n\}$, $b = \{b_1, \ldots b_m\}$. A short probability table for this case is:

	$X_{b_1} \cap \ \ldots \cap X_{b_m}$
$X_{a_1} \cap \ \ldots \cap X_{a_n}$	P_1
$-(X_{a_1} \cap \ \ldots \cap X_{a_n})$	P_2

(f) Thus P_1 is the probability that $b_1, \ldots b_m$ all occur, given that $a_1, \ldots a_n$ all occur. Likewise, P_2 is the probability that $b_1, \ldots b_m$ occur given that not all of $a_1, \ldots a_n$ occur! A brief notation for this table is ordinarily employed:

	Probability of b
a	P_1
$\bar{a}$	P_2

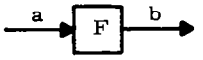
Here, we say that P_1 is the probability that b occurs given that a occurs, and P_2 is the probability that b occurs if a does not occur. When we use this terminology, it is to be understood that we are just using a shorthand terminology for what is said in the sentences above marked with a dagger (†). Finally, in many cases, we shall want P_2 to be zero. Then we shall use the even shorter table:

	Probability of b
a	P_1

and it will be understood that the probability that b occurs given that a does not occur is zero. This is the only case when leaving off part of a chart allows us to deduce something about one of the entries in the part of the chart which is deleted. Ordinarily, if part of a chart is deleted, it simply means we are not interested in those entries in the deleted part.

Function

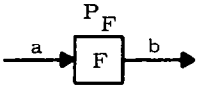
Definition: — A primitive function F = (a, b) along with its complete probability table is called simply a function. A function is sometimes denoted by the symbology:



If the short table

	Probability of b
a	P_F

is what is being used, then the function may be denoted by



Probability of Output

Definition: — In the short table above, P_F is called the probability of output of the function F. Thus, P_F is the probability of b, given a.

Primitive Array

Definition: — Let $t_1, t_2, \ldots t_n$ be the times associated with the output states of some sequence of primitive functions $\mathcal{A} = F_1, F_2, \ldots F_n$, each of which is subdivided in some pertinent way. Then $\mathcal{A}$ is called a primitive array if:

1. $t_1 \leqq t_2 \leqq \ldots \leqq t_n$
2. Every function F_j, whose output state occurs earlier than t_n has each element[1] of its output state occurring as an element in the input state of at least one of the other functions in $\mathcal{A}$.

Roughly speaking, any sequence of primitive functions where all the output states occur at the same time for one reason or another, is by definition a primitive array. But if two or more output states of a sequence of primitive functions occur at different times, then that sequence is an array only if all the elements of the earlier output states occur as elements in the input states of some of the other primitive functions. Thus, the only primitive functions in an array whose output states do not "feed into" other primitive functions, are those whose output states occur at the latest time. Notice that nothing is said which indicates that the elements of input states have to come from the output states of some of the other primitive functions. Our only restrictions are on output states.

Example: — Let $\mathcal{A} = F_1, F_2, F_3, F_4, F_5, F_6$ be a primitive array where: $F_1 = (a_1, b_1)$, the time associated with b_1 is t_1,

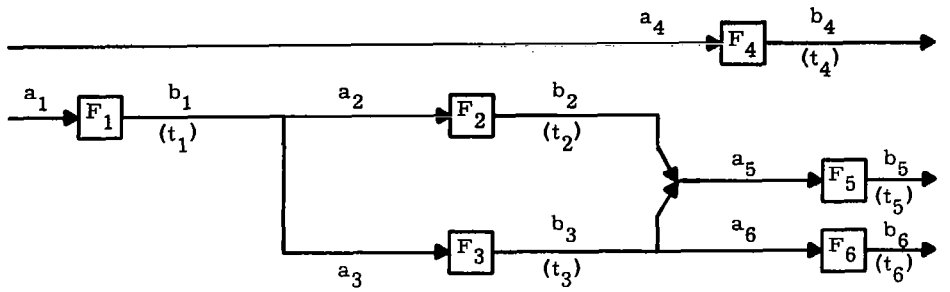
$F_2 = (a_2, b_2)$, the time associated with b_2 is t_2,

• • • • •

$F_6 = (a_6, b_6)$, and the time associated with b_6 is t_6.

[1] An "element of a state" which has been subdivided into other states, is any of the states in that subdivision.

Suppose that $t_1 < t_2$, $t_2 = t_3$, $t_3 < t_4$, and $t_4 = t_5 = t_6$. This array might be represented diagramatically[1] as follows:

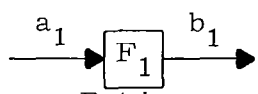


We have represented $\mathcal{A}$ in such a way as to indicate that $b_1 = \{a_2, a_3\}$, $a_5 = \{b_2, b_3\}$, and $b_3 = a_6$. With these final provisions, then $\mathcal{A}$ is indeed a legitimate primitive array.

Isomorphic Primitive Arrays

We may define isomorphism between two primitive arrays $\mathcal{A}$ and $\mathcal{A}'$ in a manner which is completely analogous to the manner in which isomorphic primitive functions were defined. The definition looks more complex only because in general there is more than one primitive function in a primitive array, and we must carefully correlate the states of each primitive function in $\mathcal{A}$ with the states of each primitive function in $\mathcal{A}'$. A precise definition of isomorphic primitive arrays may be given as follows:

---

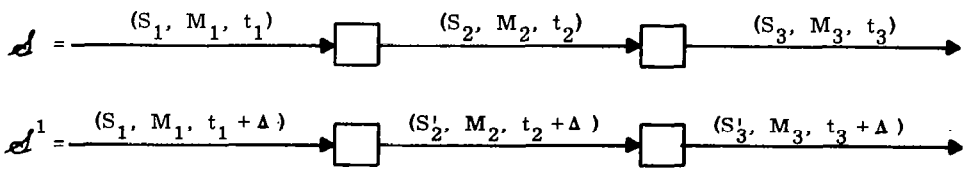
[1] Notice that in the diagram alluded to, symbols such as 

occur, which look suspiciously like our symbol for a function. But in our discussion to this point, we have spoken only of primitive functions, and indeed have said nothing about any of the things (such as sample spaces or probability tables) which are required to bring in the notion of function. Later on we shall give a definition of array, in which symbols of this sort occur and are intended to be functions. Until that time, however, we shall

Definition: — Consider two primitive arrays $\mathcal{A} = F_1, \ldots F_n$, and $\mathcal{A}' = F'_1, \ldots F'_m$, with appropriate subdivisions in the input and output states of each F_i and each F'_i. Then $\mathcal{A}$ and $\mathcal{A}'$ are isomorphic if n = m, and there is a one-to-one correspondence G between the elements of the subdivisions in $\mathcal{A}$ and those in $\mathcal{A}'$ such that:

1. For every F_k in $\mathcal{A}$, if b is a state in the input (output) state of F_k, then G (b) is a state in the input (output) state of F'_k.
2. For every F_k in $\mathcal{A}$, if (S, M, t) is a state of F_k (either in the input or in the output state) and if $G\big((S, M, t)\big) = (S', M', t')$, then M = M'.
3. There is a real number Δ, such that if (S, M, t) is any state in $\mathcal{A}$, and if $G\big((S, M, t)\big) = (S', M', t')$, then $t' = t + \Delta$.

Example: — The following two primitive arrays are isomorphic:



Probability Tables for Primitive Arrays

With each primitive array $\mathcal{A}$, we associate a class of sample spaces. Each sample space X associated with $\mathcal{A}$ is a finite set of primitive arrays $\mathcal{A}_1, \ldots \mathcal{A}_n$, each of which is isomorphic to $\mathcal{A}$. Thus the sample space

---

use the symbology $\xrightarrow{a_1}\boxed{F_1}\xrightarrow{b_1}$ to indicate a primitive function, or if you will, a function without its probability table. This sort of diagram is useful because it helps us keep track of the input and output relations between the primitive functions in $\mathcal{A}$.

for a primitive array is no longer a set of primitive functions (as was the case for the sample space for a primitive function) but rather is a set of primitive arrays. The probability measure, p, on X is defined by the number of elements in the subsets of X. Thus, as before, if Y is a subset of X, then $p(Y) = \frac{m}{n}$, where m = number of primitive arrays in Y and n = number of primitive arrays in X. Again notice that $p(X) = 1$.

Now the probability table for a primitive array is exactly analogous to the probability table for a primitive function. A slight change (but a natural one) occurs in the definition of the collection Γ of subsets of X. Here, we define Γ as follows:

1. Γ contains X_a, for each state a, in any subdivision of any primitive function in $\mathcal{A}$.

2. Γ is closed under $\cap$, $\cup$, and $-$.

3. Γ contains X.

From this point on, the probability table is defined exactly as it was defined earlier for the primitive function.

Array

Definition: — A primitive array together with its complete probability table (over an appropriate sample space) is called simply an array.

Component Arrays

Definition: — Let $\mathcal{A}$ be an array with a sample space X. Suppose $\mathcal{A}$ is the sequence of primitive functions $F_1, F_2, \ldots F_n$. Let $\bar{\mathcal{A}} = F_{i_1}, \ldots F_{i_k}$ be a subsequence of $\mathcal{A}$. Then $\bar{\mathcal{A}}$ is a component array of $\mathcal{A}$ if $\bar{\mathcal{A}}$ is itself an array, and if the sample space, $\bar{X}$, for $\bar{\mathcal{A}}$ is defined as follows:

For each array $\mathcal{A}^j = F_1^j, F_2^j, \ldots F_n^j$ in X, we define an array $\bar{\mathcal{A}}^j$ as the subsequence $F_i^j, \ldots F_{i_k}^j$ of $\mathcal{A}^j$. The set of subsequence arrays generated in this way is the sample space $\bar{X}$.

This condition on the sample space of the component array guarantees that the probability tables associated with the component array are consistent with the original array $\mathcal{A}$.

Component Function

Definition: — A component function of $\mathcal{A}$, is a component array of $\mathcal{A}$ which contains only one function.

Now that we have the notion of component function, we may speak of an array as a sequence of functions rather than as a sequence of primitive functions.

Partitioning/Adding

Definition: — Let $\mathcal{A}$ be an array with n functions $F_1, \ldots F_n$. Let $\mathcal{A}'$ be an array whose sequence of functions may be divided into n disjoint subsequences of functions $\mathcal{A}'_1, \mathcal{A}'_2, \ldots \mathcal{A}'_n$. Then $\mathcal{A}'$ is a partitioning of $\mathcal{A}$ if:

1. All the $\mathcal{A}'_1, \ldots \mathcal{A}'_n$ are component arrays of $\mathcal{A}'$.
2. The array inputs [1] of $\mathcal{A}'_i$ contain all the states in the input state of function F_i in $\mathcal{A}$ (this being true for each i = 1, ... n).
3. The array output [2] of $\mathcal{A}'_i$ is equal to the output state of function F_i in $\mathcal{A}$ (for each i = 1, ... n).

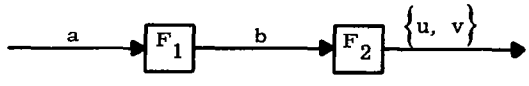
---

[1] The array inputs (of an array $\mathcal{A}$) are simply all those states in $\mathcal{A}$ which occur in the input subdivision of some function in $\mathcal{A}$, but not in the output subdivision of any function in $\mathcal{A}$.

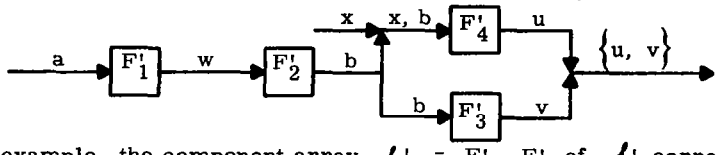
[2] Array outputs are defined in an analogous way to array inputs.

Intuitively, we may think of a partitioning of an array $\mathcal{A}$ as being another array $\mathcal{A}'$, which is gotten by "cutting up" each function of $\mathcal{A}$ into a bunch of functions. This cutting up process, however, must take care to preserve the states in the input and output of the function (although it is permissible to introduce new input states).

Example: — Let $\mathcal{A}$ be the following array:



Then a partitioning of $\mathcal{A}$ might be the following array $\mathcal{A}'$:



In the example, the component array $\mathcal{A}'_1$ = F'_1, F'_2 of $\mathcal{A}'$ corresponds to the function F_1, and the component array $\mathcal{A}'_2$ = F'_3, F'_4 corresponds to the function F_2. The array inputs of $\mathcal{A}'_1$ are the same as the inputs of function F_1 (likewise for the outputs). The set of array inputs of $\mathcal{A}'_2$ contains the inputs of function F_2, and the array output of $\mathcal{A}'_2$ is equal to the output of function F_2. Therefore, $\mathcal{A}'$ is indeed a partitioning of $\mathcal{A}$.

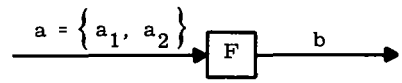
We shall want the probability tables for a partitioning $\mathcal{A}'$ of $\mathcal{A}$, to be consistent with the probability tables for $\mathcal{A}$. We may ensure this consistency by placing conditions on the sample space for $\mathcal{A}'$, just as we placed conditions on the sample space of a "component array," when we defined it. We shall assume that such conditions are placed on the sample space of $\mathcal{A}'$ as part of the definition of a partitioning. Thus for a partitioning $\mathcal{A}'$ of $\mathcal{A}$, it will always be the case that the probability tables of $\mathcal{A}'$ are consistent with those of $\mathcal{A}$.

Convention: — If $\mathcal{A}'$ is a partitioning of $\mathcal{A}$, we shall sometimes use the phraseology: "$\mathcal{A}$ is obtained from $\mathcal{A}'$ by adding."

Three Special Types of Functions

There are three important types of functions which are distinguished by their probability tables. We shall adopt a special notation for those functions which employ these tables. This notation will be used in our diagrammatical representations of arrays in order to circumvent the necessity of writing out these tables whenever these functions occur in an array.

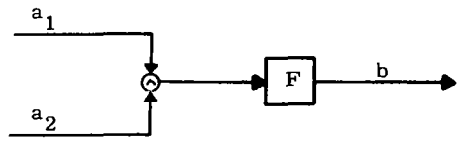
The AND Table: — Consider a function F:



If the probability table [1] for F is:

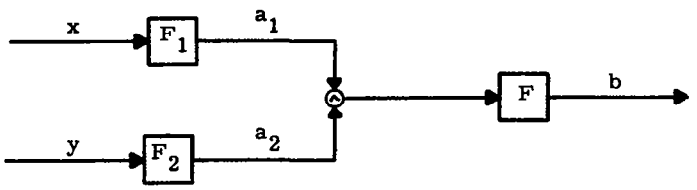
Given:	Probability of b
a_1 and a_2	P_F
a_1 and $\dot{a}_2$	0
$\dot{a}_1$ and a_2	0
$\dot{a}_1$ and $\dot{a}_2$	0

then the function F sometimes is denoted by:



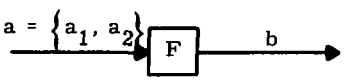
[1] In this probability table, the labels a_1 and a_2, are supposed to stand for $X_{a_1} \cap X_{a_2}$, and the label $\dot{a}_1$ and a_2 is supposed to stand for $(-X_{a_1}) \cap (X_{a_2})$. Likewise the label "probability of b" is supposed to stand for X_b. The use of these new labels above, make for a quicker interpretation of the entries in the table.

The symbol $\wedge$ is called the AND symbol, and the table is called the AND table. Consider for example the array:



This array may seem to convey the idea that somehow the outputs of F_1 and F_2 are "joined" so that they can be fed into F. Actually, what this symbology tells us is that the input to F is precisely the state $\{a_1, a_2\}$, and the probability table for F is the AND table. [1] No "joining" function is implied.

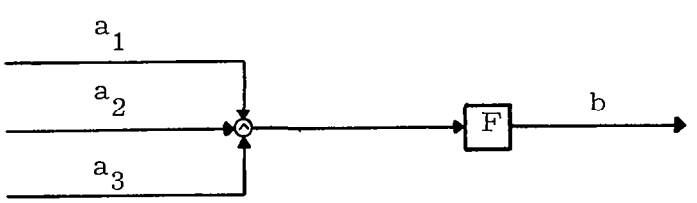
The Exclusive OR Table: — Consider the function F:



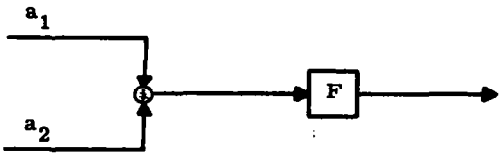
If the probability table this time is:

Given	Probability of b
a_1 and a_2	0
a_1 and $\dot{a}_2$	P_F
$\dot{a}_1$ and a_2	P_F
$\dot{a}_1$ and $\dot{a}_2$	0

[1] The AND table above is for an input state with two component states. The table may be easily generalized to input states with three or more component states. The notation would not change. Thus, in a pictorial representation, the AND table for three components would be implied by:



then the function F will be denoted by:

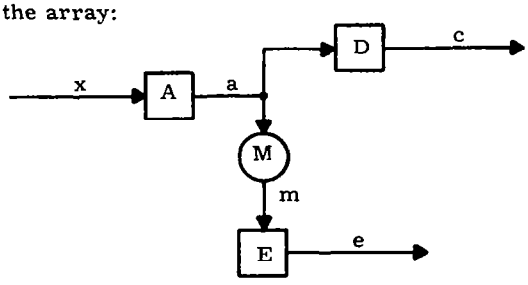


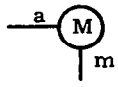
The symbol ⊕ is called the exclusive OR symbol, and the table is called the exclusive OR table. This symbol tells us that the input to F is precisely $\{a_1, a_2\}$, and the probability table for F is the exclusive OR table.

The Monitoring Table: — Consider the function C = (a, b). Suppose this function is accompanied with the following probability table:

	Probability of b
a	0
$\bar{a}$	P_C

When such a probability table is associated with (a, b), then (a, b) is called a monitoring function. The table is called a monitoring table. In a pictorial representation of an array, a monitoring function will be denoted by a large circle. Thus in the array:





denotes a monitoring function.

Roughly speaking, M responds when the output of A fails to occur.

III. USE OF THE CALCULUS

Thus far we have not employed the term system. That is because system is simply a term of convenience within the calculus. Its most frequent use will be outside of the calculus when making application of it. Indeed, the use of the term system in the title of this paper is an extramathematical usage.

Within the calculus it is useful to have a term to refer to the "most comprehensive" array that will be considered in a given discourse. Thus, suppose $\mathscr{A}_1, \ldots \mathscr{A}_n$ are all the arrays under consideration in a given discourse. A very typical situation is that one of these arrays (say $\mathscr{A}_1$) is a "parent" array to the others. By this we mean that the other arrays may be divided into two classes C and C' where each $\mathscr{A}_i$ in C is a partitioning of $\mathscr{A}_1$, and each $\mathscr{A}_j$ in C' is either a component of $\mathscr{A}_1$ or a component of some $\mathscr{A}_i$ in C. We say that this parent array is the most comprehensive of all the arrays under consideration. In a natural way we may associate a unique function (a, b) with this parent array. The input state a of this function is the earliest of the array inputs of $\mathscr{A}_1$, and the output state b is the array output of $\mathscr{A}_1$. We use the word system to refer both to the function (a, b) and to any partitioning of it.

System is used in engineering and in everyday communication with a wide variety of connotations. In fact, one might suspect that the number of connotations is somewhat greater than the number of users. In a situation in which the calculus is being applied, it is suggested that the use of the word system to refer to real world objects be restricted to application to a collection of objects that may be set in correspondence with the parent function in the discourse.

The calculus that we have presented, then, is intended to be useful for talking about real world systems. The systems to which it may be applied will be discrete systems or they will be systems which may reasonably be treated as discrete systems. Thus the calculus may be used only to talk about real world systems all of whose system outputs occur at the same time. Our principal interest is in development cycles.

In the introductory section of this paper, and in the above paragraph, we employed the term "development cycle" with the expectation that it would be understood in the everyday sense. Without a calculus, that is about the best that we can hope for. We are now in a position to identify more precisely how we wish to use the term.

A development cycle is a discrete system whose input is a Primitive Need Statement [1] and whose output is a collection of means for implementing an operational system [2] that will satisfy the need which occasioned the Primitive Need Statement. Not all processes in the real world that we would like to call development cycles are discrete, and thus not all development cycles in the everyday sense of the phrase match the definition above. However, we believe experience will show that virtually all development cycles (in the everyday sense) may be treated as discrete systems for the purpose of designing and controlling them.

Once a development cycle that is of practical interest has been described as a system by identifying its input and output states, we may employ the calculus as an aid to determining an appropriate basic strategy for carrying out the development process. Thus, a development cycle can be partitioned to define a prime function array [3] in a manner that precisely identifies a chosen strategy. [4] A strategy that is described in this manner can be presented for public inspection and, as the result of such inspection, may be corrected,

---

[1] By Primitive Need Statement, we mean the first verbalization which has the effect of calling attention to a real world problem which requires a new system for its solution (ref. 2 , page 18).

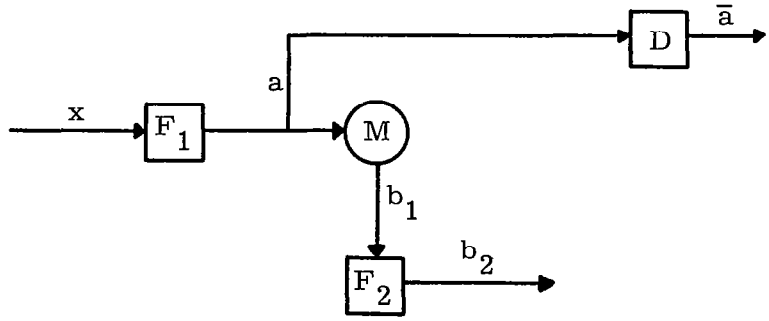
[2] A man-made system which is built to satisfy a need in another system is called an operational system.

[3] A prime function array is an array in which the probability of the array output (given the first array input) is equal to the product of the output probabilities of all the component functions. Recall that an "array input" is a state in the array which does not occur as an output of any of the functions in that array. Recall also that states which occur at the same time may be considered as a single state or as multiple states, whichever is the most convenient.

[4] By a strategy we mean here a sequence of steps for carrying out the development process which can be justified where justification is in terms of cost of

accepted, or rejected. An example of a basic strategy for carrying out the development of complex aerospace systems is presented in Part A of this report.

Not only does the calculus permit the precise description of a strategy in terms of the prime functions essential to the development cycle, but it also permits the elaboration of such a prime function array to provide for high probability of success of the development process being described. To achieve such a goal, we use monitoring functions to build additive loops. Thus, a monitoring function, M, is often used in an array as follows:

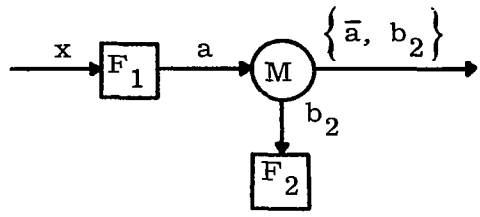


Here $\bar{a}$ denotes the same state as a, except that $t_{\bar{a}} > t_a$ (see footnote 1). We interpret the purpose of the monitoring function in this array as follows: M responds only if the state a fails to occur. If a does not occur, M produces b_1, which is an input to function F_2 which in turn produces output b_2. The probability that $\bar{a}$ or b_2, or both occur, is greater than the probability that $\bar{a}$ occurs, if P_M and P_{F_2} are both greater than zero. The role of M in the array is defined precisely only by the probability table associated with M.

development and in terms of the quality and cost of use of the system that is produced by the development cycle. See Report I, A Simple Model of a Man-Machine Development Cycle.

[1] The function D is inserted to preserve time relationships such that $t_{\bar{a}} = t_{b_2}$.

An abbreviated notation for the above array is sometimes employed:



In such an array, the set of functions $\{M, F_2\}$ is called a first-order additive loop.

By means of additive loops we can provide for "management" to ensure the high probability of the success of a development cycle. Chapter 5 of Part A suggests some of the ways in which this may be done.

By the manner in which the concept of function and array are defined in the calculus it is articulated with probability calculus such that one may readily derive a system description in terms of a probability equation given a system description in terms of the calculus. One may also readily translate a system description in terms of the calculus into a PERT description. In this translation, each function becomes a PERT activity, and time relationships are preserved.

Any attempt to recapitulate the rationale underlying the selection of the specific basic concepts and syntax which make up the calculus presented above would be both incomplete and tedious. It would be incomplete because much of the rationale is difficult to retrieve. The calculus was employed in its earliest form as an informal working tool; initially there was no intention to formalize it. It evolved as a working tool over several years as it was used to help solve a wide variety of system problems. By the time the decision was made to formalize it, the calculus was well shaped as an intuitive method and the many specific motives underlying it, like most evolutionary forces, were no longer identifiable. The best that can be done now is to test whether the calculus can be used to begin the task of describing the system development

process; one attempt is presented in Part A. Whether or not the adjustments made over the years of use have indeed generated a calculus that is broadly useful (or which is amenable to modifications so that it will be useful) can only be determined if others attempt to use it as an aid to solving real problems. No amount of rationalization will make it any better than it is.

Serendipity Associates
Chatsworth, California, October 1966.

REFERENCES

1. Air Force Systems Command: Configuration Management During Definition and Acquisition Phases. Air Force Systems Command Manual, Systems Management, AFSCM 375-1, 1 June 1964.

   AFSCM 375-2 — System Program Management Surveys and Industrial Management Assistance Surveys. 25 June 1963.

   AFSCM 375-3 — System Program Office Manual. 15 June 1964.

   AFSCM 375-4 — Systems Program Management Manual. 16 March 1964.

   AFSCM 375-5 — System Engineering Management Procedures. 14 December 1964.

   AFSCM 375-6 — Development Engineering (DE). 14 August 1964.

2. Asimow, M.: Introduction to Design. Prentice-Hall, (Englewood Cliffs, New Jersey), 1962.

3. Department of Defense: Initiation of Engineering and Operational Systems Development. DOD Directive Number 3200.9, July 1, 1965.

4. Department of Defense and National Aeronautics and Space Administration: DOD and NASA Guide PERT COST. Joint Services, Office of the Secretary of Defense and National Aeronautics and Space Administration, June 1962.

5. Goode, Harry H.; and Machol, Robert E.: System Engineering: An Introduction to the Design of Large-Scale Systems. McGraw-Hill Book Co., Inc., (New York), 1957.

6. Gosling, W.: The Design of Engineering Systems. John Wiley & Sons, Inc., (New York), 1962.

7. Whitehead, Alfred N.: Mathematics as an Element in the History of Thought. In The World of Mathematics, Vol. 1, James R. Newman, Simon and Schuster, (New York), 1956, pp. 402-416.

"The aeronautical and space activities of the United States shall be conducted so as to contribute . . . to the expansion of human knowledge of phenomena in the atmosphere and space. The Administration shall provide for the widest practicable and appropriate dissemination of information concerning its activities and the results thereof."

—NATIONAL AERONAUTICS AND SPACE ACT OF 1958

NASA SCIENTIFIC AND TECHNICAL PUBLICATIONS

TECHNICAL REPORTS: Scientific and technical information considered important, complete, and a lasting contribution to existing knowledge.

TECHNICAL NOTES: Information less broad in scope but nevertheless of importance as a contribution to existing knowledge.

TECHNICAL MEMORANDUMS: Information receiving limited distribution because of preliminary data, security classification, or other reasons.

CONTRACTOR REPORTS: Scientific and technical information generated under a NASA contract or grant and considered an important contribution to existing knowledge.

TECHNICAL TRANSLATIONS: Information published in a foreign language considered to merit NASA distribution in English.

SPECIAL PUBLICATIONS: Information derived from or of value to NASA activities. Publications include conference proceedings, monographs, data compilations, handbooks, sourcebooks, and special bibliographies.

TECHNOLOGY UTILIZATION PUBLICATIONS: Information on technology used by NASA that may be of particular interest in commercial and other non-aerospace applications. Publications include Tech Briefs, Technology Utilization Reports and Notes, and Technology Surveys.

Details on the availability of these publications may be obtained from:

SCIENTIFIC AND TECHNICAL INFORMATION DIVISION

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Washington, D.C. 20546